RNB ZR CS 198 8:30-9:50 a.m. ET
September 19, 2024
>> INSTRUCTOR: For sets, we also use parentheses. And there's an important distinction between these three types of data subsets and this one . Very important distinction. The dictionaries are (Indiscernible) it's quite different to this. We call them characteristic . This two. There are sequences. These two. Sets -- let me do here -- sets are different from these two in that sense. Sets have no sequences . When we say that something is a sequence, it's because it exists in order, and sets do not have order. [word?] Analysis totally different to this, totally different to this, totally different to this. It's mapping. When you have key coccode -- codon and.[word?]. Collections of these things. Call Peter column A. John column B. I cannot associate grades with names as an example. This language, Python, is very syntactic.
What that means is unfortunately, this is my personal opinion, you have to watch out where do you put the quotes, where do you put the comma. It's very syntactic in that regard. You have to be very careful in that regard. The only way to handle this is to do the exercises in the book. It's very difficult I would say copper for me at least, to remember all the details of the syntactical so I can see why the book is a great tool. You know, you see cough you run it because organicities answer is. You're gonna repeat etc. But conceptually, despite all the syntactical knowledge, these are the things to remember. About this collections of data. What are the important operations that you can perform on that data? The first operation is membership.
To be able to test different elements is in this set or not. And the operator used in Python for that is two letters. IN For in . That's membership. The other important operator is the length operator LEN, and this tells you the number of elements in the collection. The other one is that when you have a certain name, you can use brackets. And here you could put an index. Let's say I check let me put a number. 17. So this is the name of your collection, whatever name you have. Black 17 is indicating that's the 17th element in that collection. Is that clear? What is the caveat about that? What is the caveat about that? The caveat is that this accessing of data by using an index is not possible always.
It depends of the type of collection that you're dealing with. This indexing is okay for lists and it's okay for tuples . So this business of indices is related to the fact that these two are

sequences . Where you have a sequence, there is order that you can index . That's not true for sets. So any time that any type of a statement that you have, it's important to share what is the perimeter of that function, and it's important to check the perimeter can be indexed. There is a generalization of that . Usually when we say indices in this case, it's because we who like to (Indiscernible) that by a number. The assumption is whatever data you're having can be viewed, and this is just a view . This is just a view of indexing of the data that you're dealing with that can be viewed as an ordered collection of boxes . And in each of these boxes, you have an element.
Then The first one is the (Indiscernible) that is the assumption, so you have a collection that can be viewed in this way . That means it can be indexed . The classical example of that is things. The lower type of collection that you can think of before these guys are the slings (sp?) Slings are the most atomic collection. I want to make sure you understand the strings can be used to create lists, create tuples, create sets, create dictionaries so it's very atomic . Strings are the typical collection that can be indexed. Now you have to be careful. Because every language creates new datatypes differently and have to look at the definition of the language in its specification and it will tell you.
This Particular type can be indexed by accessing elements with an integer. What happens to be that the generalization of that is when you're dealing with dictionaries, it's not that they can be indexed, but they can be accessed by keys . The way to understand this is if you have chiasm example, created column A for grades. Jay Column F. Emmy column C. Suji: B etc. so this is collection of key value pairs. The keys are the names. The value are the grades as an example. This is a collection of grades. Then there's a weigh in we will look at examples of saying grades accessed by the key . As an example, the grades is the name of the collection . You can write grades and right here Emmy . This is saying that collection, that dictionary that has been formatted in this way, you can access a particular element by specifying the key as a parameter.
In This case grades Emmy would be you assign that to a value to say call G . What is the value of G? After that statement from this example? G is C. Grades is the name of this. Emmy is the key. Grades MER C see you assign it to G value C. Is that clear? The point that I'm making here there are ways to access data depending on the type of collection by keys in this dictionary. But do you have things like lists and tuples? You can access them by index. That is the difference. Okay? There is a name for this generalization where you have access either by keys like here or by index like there. The name for this type of collections are iterable . That means you can iterate over the collection. Why are we pushing on this? It's because when you start doing these exercises, you do an example for strings, example for tuples, example for sets.
There Are over whole bunch of rules and nothing to do is create a table yourself that checks if I have a list, how do I check membership? How do I check the length? Can I index that list? The answer is yes. But you go to sets, you can do this, you can do this, but you cannot index the set. So you re-create your own table listing all of these and putting checkmarks for the type of operations that you can do (Indiscernible) to put together what you're (Indiscernible) examples I see a list, brackets, lists of integers can be lists of

something else . Here this is a list of what are the subjects? (Indiscernible) Strings . This is a mixed list . The string separated by commas . What is the commonality here? What am I using brackets and commas? Yes?
Types of operations that you want to do. Membership, length and indices. Given this list, because you have brackets, A is the value of this list and this is the same as 2.5 in A . This is the operator that is checking that it's 2.5 is in A. What is the value of this expression? Is this false for this example or is it true? True . What about this one? Printed three names. What is the value of this expression? Three names? False . This will print true. This will print false . I want to emphasize this is applying the discussion in the lecture that there are these expressions that have values, and that's what this is doing. You are evaluating expressions inside and printing them using the membership operator. The length of A. What should be the answer here? Length means how many elements an object has.
This Is the object. How many objects -- elements the subject has? Look at the commas . It has 123 elements. This one. A which is the subject . This block is zero, so what is the understanding on this? A is this. It's sequence. And so list. This A with brackets means single element in that list. Which one is that? Can you tell may the answer?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: What is the answer?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: Yeah. What is your answer? This is A? Single element of A? What is that? Yeah? One? Zero element of this list is this guy. This is value one so this would be one. Raise your hand if you get this. What about this one? Remember the trick to get all this established always every way the deepest parentheses first. The deepest out. Here, this is in this case the deepest parenthesis is length A -1 with this brackets. What is length A- one? What is this value here? Forget the A for a second. What is this value here?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: What?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: What is it? Two? This is two? The length of A is 3 - 1 is to so now A evaluated at two, that means it gets me there. Second element on that list, which is 2.5. This one is kind of tricky. A -1. That's kind of tricky. There's a convention in Python that this will give you -- come on, there's some people that claim that they know Python . What is this? Yeah?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: The last element for the list. It's funny that it's -1. This allows you to check your answer. This is very useful now. This function LIST takes what is given as an element and makes a list out of it. Again, the way to link this is the argument here is a string . The string containing ABCDE . This is one object. This string is only one object. There's one string. Okay? When you apply the function LIST to that one object, that function LIST transforms that object into a list of its components. ABCDE is six string of five characters . When you apply LIST to it, what this is going to do is it's going to create a list for you consisting of each of those elements . In this case, each of these characters. The subject is

completely different -this object is completely different from the initial object. The initial object is one string. .
When You apply this function, now you have a list. With five objects. Which object is a character . This is very useful . Another very useful thing is that this list can be nested . Here you have this is a list has two commas here so this list has three elements. This is one element. This is another element . And this is another element. This list has three elements. But the single element is a list itself. Is not clear? This list contains 123 elements, and the single element, which is the first one you want to think, is a list by itself. Now you apply the length of the (Indiscernible) to this object . This will be answering you three. Why? Because this outermost list contains 123 objects.
So when you apply the length operator to this, if it's nested to the answer that you get is the outermost number of elements, so in this case three. Okay. Now when you have this nested list, the nice thing is you can index them by using double or triple indices that you want. Let's look at the example of double indices. Right here in this example, you have a list . This is the outermost brackets. That's a list. This list contains one element to two elements, three elements, four elements. The first element or the single element is a list that contains two elements. The first element contains -- excuse may the first list contains two elements. The first list contains two elements. So now you write down you assign this to the nested list and you write down two and one in brackets. This is telling you you access the -- what this means this two?
The Second element in this list, and when you get the second element of that list access the first element of that. What is the second element in the list? This is the first element because second element, third element. The second element is this list here and this list, that's the second one here. Now we are going to focus on that one and that one . You want element one so this is the single element for this and this is the first element of this and the answer is.[word?] Which is here. These are the four assistants for the classroom. Okay? A list is mutable. That means you can change the elements in the list. For example here, just to make sure that this is clear, this is a list assigned to the value A so the type of A list, now you want to use this A function to -- F function to print it in a precise and specified way. A and D parentheses DOA is the location of the subject and the value of that object . Now suppose now you access the single element of value A.
The single value is what you want and suppose you take the number five and you store it in that location. This is what this is doing for you. What this is doing is replacing this -- replacing this by five and now you do the same statement you will see that you get instead of 123 you would get 523. What are the elements here that you have to put together if you've never seen this? The elements here is you need to know what is the type of object you're dealing with . In this case it's a list. Because it's a list, you know that you can index the element of that list. That means you can access any element on that list . To access that, you use this mechanism of brackets of the index and at the time you can index whatever and you can change it or you can read it etc.
We have several caveats here. For example, you're looking at this particular example . A now is a type of string because this is on the right-hand side. This is string 123 assigned to

A. You want to bring that ideal way which is the location for that and the value of that and then you would get the answer this is the idea and the value 123 but now it complains. I was accessing the single element and I was putting I wanted to store it in five. Here's the one I wanted to put five. I would not have it. So this is contradicting what I told you. I just told you that you have a list. I told you you can index the elements of a list. Here you have 123. What is 1 to 3 on that first time it's not a list. You can make it a list and then index them.
The Idea here is you always have to be careful what is the type of object you are dealing with. A string is a string . That cannot be indexed. You want to index it, you can use the previous trick for transforming that into a list and now you can index that list so that's what's happening. Another important operation for these things is you want to add stuff to it. Depending on the collection and given operations to add stuff to your objects. So you have a list . You have a method . You have something like this in Python . That means whatever you write is a function that can be applied to that list . In this case you want to append to a list, and it has some brackets to pull the particular articles.
The Important thing to understand here is that the specification of this list, it tells you directly this will take here a single element . This is seeing you want to use this? The argument should be a single element. Let's try it out. 123 (Indiscernible) assigned to X. Now you say X (Indiscernible) but you want to append to this list X the element four so four will be added to this. All I'm saying is that this argument has to be a single element. So you printed and now four gets added to the list. There's something very nice . This part is nice in the language is that you can actually append an entire list to a list that you already have. Up to this point, X contains the list 1234.
Now I would like to append to this another list which is 45. So I would like to attach this to this . The trick is that this is saying this has to be a single element. So you say X dollar append parentheses four five, this is a single object. So this is okay . This is a single object. If you were going to ignore these brackets, this would not work because append will require here to have a single object but you put the brackets, that's a single object. Now notice what happens here. This 1234, you attached to it another object which is 45 as a single object. This is not going to be discreet increases or anything like that. So append is nice. There's something actually really useful in massive data mining which is this operation called extend . Extend is a more sophisticated type of append.
You want to extend a list by something, and in general, what this is telling you is that you can put here a single any interable type. You can put in a simple string here or you can put here a list or you can put here a tuple or a cell in the dictionary so this is very generic here as long as you put an argument that is one of these things that can be iterated . What this extend does this goes inside whatever you put here is an argument and takes this stuff inside and splits it out for you. Let's see the example here. You have a list 123 and you would like to extend that by this object here, which is 4567. Extend is different to append . Append comes in on one single object. Extend takes this cog goes inside cut takes every single piece and attaches it to the existing list here and this is the application of this. Moral of the story. Append and extend are related, but extend is a lot more sophisticated.

It Takes a complicated structure that you want to attach to something existing, and it goes inside of it and it breaks it down and creates this larger, nice object and you have to be careful where the errors are. Here another nice example. You have the same 123 and you want to extend it by the string ABC but you are using extend . This is going to go inside ABC and look at each individual component and it's going to attach it to this so you can create lists out of existing objects. That's the beauty and actually the power of this. Many times in data processing, you would like to delete an element, and you have a list to delete an element. You use this function called pop so you're going to pop something from the list. You Specify the index of the element you would like to remove and remove it. When you don't specify the index, then the last element is removed by default. Okay. Now we pass to another type of collection which is called a tuple, and you will see regarding these lists it's so important to see the differences between possible lists. These are called tuples. The difference between tuples and lists our tuples are immutable. Once you create a tuple, it cannot be changed. Fixed factors. You see where this fits in general data science. You can think that you have a record of information, and each of these objects has a name . Maybe a name . Again just the last time we did an example with the name, grade, maybe age, weight, etc. This is a record of information about let's say a particular individual. And you can think of that Peter A 25 etc.
That whole collection you can think of that as a tuple. This is very, very, very, very basic and important data construction to have tuples. When you specify how many pieces your tuple has, that is called the schema for your data and we'll be using that when we get to relational databases in three weeks or so. Tuples are fixed length objects. Excuse me, fixed (Indiscernible) this is a tuple. You put parentheses. This tuple has two pieces. One is (Indiscernible) and one is a string and you print it (Indiscernible) . There's another way to create tuples. There's a function called tuple. You write TUPLE parentheses and inside you put whatever you want and that creates a tuple.
What would be the output after this assignment our what should be the output here? This is good way to study by the way. Parentheses two comma A, but A has some so it should be two comma quote A quote A What about here? What happens if you write a tuple like this? How many elements does this tuple have by the way? How many elements this tuple has? One. The outermost is one object. And this is the point about using a tuple that what you put inside the argument, the specification, is one object that can be iterated. Notice this last transformation from here to here. The argument to this tuple is what? It's string . This is the argument to this tuple is a string. How out of that string do you get a tuple which is something like this? A record of information.
When You apply tuple to that string, this is what happens. The individual components are spit out and now from here on, you can treat this whole thing as a tuple . On the first position it has this value. On the second position has this value etc. You may be asking yourself, what is the difference between strings and lists? A string is quite different to tuples. This is a string. This is a tuple. Yes, this tuple is obtained from the string, but using the tuple function. This cannot be modified. And you can become very creative with tuples. For example, look at this example. By the way, this parentheses sometimes you may omit

them . It's better not to omit them. The purpose of this exercise is not that. The purpose of this exercise is from the right side you have something string comma two. On the left-hand side, you have A comma B which are two volumes.
Here Is this assignment statement so this is called (Indiscernible) in general when we have spoken of assignments, we have something on the right-hand side being assigned to one value on the left. Here this is telling you that you can use this mechanism of making multiple assignments at the same time. Here, you have three and two separated by commas . These are two things. On the left-hand side you have HEY, MAC B. So after you say print, this value A comma B, you get three comma two. You may not be paying attention to this detail very carefully. The purpose of this example is not to tell you how to print three comma two. That's not the mportance. That's reality.
The Purpose of this is to show you that the language allows you to assign several things to several things as long as they have the same number of items. At the same time . You could achieve the same thing by first assigning three to A and then assigning two to B and then doing it. The language that allows you to do this simultaneously (Indiscernible) and now when you do that ConA look you can do things like this. BA It is assigned to AB . For this, the value of B is assigned to A and the value of A's assigned to B . This is flipping the two. With one statement. Sets. Sets don't have order, but for these languages to be able to access elements in a set, you have to use your mechanism internally which is called hashing . Can anybody tell me what hashing is? Yeah?
>> STUDENT: Framing the value with a function based on some mathematical operation.
>> INSTRUCTOR: Wow . Repeat again.
>> STUDENT: Assigning value to a function or object --
>> INSTRUCTOR: No because assigning a value to an object . To a collection of objects.
>> STUDENT: Based on mathematical operation.
>> INSTRUCTOR: Based on mathematical operation. Are you running for office? That's a very good description . Hashing is assigning value to a collection of objects. What kind of value? It's important. What is the type of value that hashing function assigns to an object? This is a potato. This is a Coca-Cola. Give me (Indiscernible) question again. When we speak of hashing, taking some set of data objects. This is a mechanism to assign to the subject a value. What kind of value?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: No . It cannot be (Indiscernible). A number . It cannot be any type of number. Must be an integer. Why? Why it must be an integer? Say it again louder please? Okay. It must be an integer because these integers can be thought of as identified buckets or positions in memory can be followed. Positions in memory . The importance of this is you have a whole URL let's say of a page . You like to assigned to that whole URL a number. Rather than that whole bunch of factors concatenated together call you would like to assignment to a number. We talk of hashing, it's important to remember . This is a way to assigned to objects integers and there are different types and there are different classes of hashing functions and some of them are not very sophisticated or simple. To understand. You want to learn more about this. You have to like prime numbers.

They all have hashing functions. So if I was going to list the names of all you guys, it would be hard for me to remember all the names. I will apply a hashing and then each of you will become a number. Then at some point in the class, and we may do this actually, we have a spinning wheel and that spinning wheel will generate out of those numbers a random number. I don't know who the person is . Mr. 175. Mr. 175 will be one of you and then you will be asked a question. It's a mechanism to treat data by mapping the data into numbers. These are hashing functions. And of course, there are certain conditions for individuals to be mapped to the same (Indiscernible) a lot about hashing functions. From your viewpoint, you just have to think that you have something like let's say ABC and you apply the hash function, now what object is this now?
This is string ABC and you apply a hash function to it. What object is this now? Hash of ABC, what is it?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: It's an integer . Now you can manipulate this integer anyway you want. So this is the way that strings are used . You don't have to understand what is inside the hash function. You just have to understand that there is a mechanism in the language that assigns to that object some integer. In this case this hash is applied to this very complicated object and this very complicated object after that is an integer. To conclude this part of the class, all these things (Microphone Interference) but you who like to build sets out of basic objects. Trusting. You can create a set by using this kind of parentheses, curly brackets if you want. Here it says 223 and you want a set out of it whereby definition sets cannot have repetitions. So when you apply this assignment and you want this to be a set call when you put these curly brackets, automatically in the language the repetitions are ignored.
So There are two 2s and only 12 on the screen. This is as I said is 23, not to 23. Sets can have things of different types. For example you said two has this element or string three string two blah, blah, blah and these are the elements of the set. Be careful. This too is not this two. This two is the integer two. There's two is the string TWO. The restriction about sets is you cannot have inside the set you cannot have a list as members of a set. So this is not good. You cannot say a set that has inside a list. Sets don't have order so they cannot be integers so you say set two bracket index . No, that cannot be done. About this is that sets can be billed from any interable collection. You just have to bind the interable to this particular function SET . What is nice is it takes whatever you put here is an argument and it creates a set out of it. Let's look at examples.
Here is a string and you put SET in front of it, this is going to create a set for you out of this. In this case, that set is LPEA. nevermind about the order. It depends on different implementations in the language but nevermind about that. The important thing is here with two P and only one P. When you apply SET to it because the particular elements of this string are separated and now you have a set consisting of all the characters that appear in that string. Look at this one. This is set and now you have 14321 and now you print it and you get 1234. Because this one is only one element Necessary. XY ectasy. X is repeated here so one element is printed. The thing to remember is this is a very powerful construction. SET Makes a set out of any interable argument that you put down. Now there

are certain things that you can do to a set. You can test if an element is in the set. You can check the length but you cannot index a set.
You cannot say give may the 17th element in your set. You can add elements to a set. For example, you have this assignment because this is this curly bracket saying this is a set. You store in A . Now you want to add an element to it. You take A, the element five and this element five gets added to the set. But if five is already in the set, that would not be added. I'll give you examples of adding more things. The last example I want to mention is this one. This is extremely powerful . You have a set and now you want to apply the update function to that set. You can put here as another argument any interable collection whether it's a string, list, tuple, another set, and what it will do is will take each element of that interable that you put here is an argument and add it to the sat. look at an example here.
We have set A with these integers and B and the second element is 3X and now you would like to update this. What happens? Anybody? Maybe this is simpler for you, the last one. You have A and you would like to update it with (Indiscernible) Z. A is that. Look what happens. This becomes that. This is that. This one is that one. B is that B. At some point, you updated that with five and the next time you updated that with AZ both this AZ is between a and C and it's added. It's a little bit thought behind this so you need to understand. You can remove elements from a set . That's very simple. The element is not in the set and you try to remove it, the language will tell you hey. you are trying to remove something that is not there. if you want to be safe, instead of removing cow you can say discard and if the element isn't there. don't give me an arrow.
I Think this is a good time to stop and we'll finish dictionaries. One thing that I would like, next week we will be assigning to you your first group project, mini project. And you -- in order for you to get ready for that, those of you that don't know how to set up your local notebook, there are no books that you can get on the cloud, but sometimes you would like to have your own local notebook so the TA will be helping you with that in the recitation. The second item is that the first notebook that you will be creating for this class will be a simulation of a random walk . It doesn't sound very sophisticated. You will (Indiscernible) and the TA will be teaching you how in about 15 or 20 lines of Python code you can simulate a random walk . What is a random walk in the (Indiscernible) now random you are going to assign them to go up, left, right or down and actually you can print your random walk.
You Are interested in developing games, randomization is a very important element in creating games so we want you to develop your own notebook with your group to tackle things of this sort and this is gonna be covered in your recitation today. Yeah?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: (Indiscernible) We will have a (Indiscernible) session if you want to gain some points. So when you finish the quiz, go to (Indiscernible) and.[name?] Is going to put the information. First go to the quiz and when you finish the quiz, you are allowed to participate in the exercise and you'll be able to gain some points. You should be doing the quiz now by the way. All you have to do is type that URL to get to samurai (sp?) . Everybody

ready? You are going to have three-minute questions so you can get bonus.
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: If you join in the chat, then you cannot participate. To join the bonus, that's the URL. It's a short link URL. It's the same as last time.
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: We will tell you.
(End Of Session)