Session Code	CODR-912-BC-004
Module	Basic
Teaching Unit	Data Structures
Learning Outcome	String, Collections: list, tuple, dictionary, sets
Resources	Teacher: 1. Laptop along with audio and video exchange 2. Notebook and Pen(To note any development from session) Student Resources 1. Laptop along with audio and video exchange 2. Notebook and Pen(To keep note of important parts in the session)
Duration	50 Mins

Structure	**Warm-up** **Pace-up Activity** **Knowledge Transfer** **Student Led Activity** **Short Quiz** **Heads up tip for next class**	2 Mins 5 Mins 10 Mins 20 Mins 8 Mins 5 Mins

Step	Say	Perform
Warm up (2 Mins)	Hi *student name*, how are you? Are you excited for the class? Do you remember the last class?	Engage with the student in conversation.
Interaction (5 Mins)	In the last class, we learnt about Conditionals: • **if** statements • **if else** statements • **elif** statements We also learned about **int, float** functions, which can be used to change data type. We also learned about **Relational** and **Logical** operators.	Revise the concepts learned in the previous class.
	Today we will learn about some Data structures of Python.	
Teacher shares the screen and open Repl.it		
Knowledge Transfer	Do you remember what variables are? Yes, a variable is storage location which holds data value. Similar to a variable, Data structure is a collection of data values. Some in-built data structures are: • List • Tuples • Sets • Dictionary We will study about them today. But before that let study about **Strings**	While talking about types of data structures, keep in mind that Strings, Lists and Tuples are mandatory to cover. Sets and Dictionary are optional topics depending upon time constraints and student's competency level.
	Strings as we have learned in earlier classes are strings of characters that we can use in either single or double quotes.	
	There are some useful functions that we can use on Strings, let's study them.	
	• **len()** gives the length of the string, Syntax: len(string) • **.lower()** returns the whole string in lower case, Syntax: string.lower() • **.upper()** returns the whole string in upper case, Syntax: string.upper()	Create a string and use these functions to show how they work

	• To connect two string variables we can use **+** operator, this is known as concatenation.	
	We can extract parts of the string from an existing string using the index of each character. The index of the first character is 0 and second is 1 and so on. **Eg.** **x=”mytoppr”** **y=x[2:5]**	
• **Stop sharing your screen** • **Ask the student to share screen and click on Student Activity 1: Strings**		
	Help the student solve all the questions. *In case the student solves all the questions too quickly, add in more questions to test use of all the functions learned till now.*	Student Activity 1: Strings
• **Ask the student to stop sharing screen** • **Share your screen and open repl.it**		
	Great so we learned a few important functions of Strings, now let’s learn about inbuilt data structures of python. First of them being List	
	As the name suggests **List** is simply a collection of different data values. These data values can be of different data types. So we can have a list of numbers, a list of names and even a list having both numbers and names.	
	Let’s see how to create a list. Just using square brackets, we can easily create a list. **a_list=[1,”new”,3.8,True]** We can even have a list inside a list. **new_list=[34,a_list,”old”]**	Emphasise that each element of this list is a different data type.
	The elements in a list are ordered, which means each element in a list has a unique address and this unique address is called index.	

	The first element of a list has index 0 and the second element has index 1 and so on. To access any element of a list we can simply use its index, this process is called indexing.	Refer to indexing in String
	So to access the second element of **a_list,** we can simply write **a_list[1]** This is called positive indexing.	
	We can also access the element using negative indexes which start from -1 denoting the last element.	Teacher Activity 2: Index of List
	Instead of individual elements we can also access subsets of the list with their index. **a_list[1:3]** will return the second and the third element of the list.	
	To add an element in a list, we have different functions: • **.append()**:it is used to add an element at the end of the list Syntax: list.append(element) • **.insert()**: it is used to add an element at the specified index Syntax: list.insert(index,element)	Use and show examples for these functions.
	To remove an element in a list we again have different ways: • **del** keyword: it can be used to delete the whole list or a particular element using its index Syntax: del list or del list[index] Eg. del a_list or del a_list[3] • **.remove()** function is used to delete an element by its value Syntax: list.remove(element) Eg. a_list.remove(2)	Use and show examples for these keywords and functions.
	Similar to **list** we have another data structure called **tuples.** It is quite similar to it, except that we can add and remove elements from the list, while we can't do the same for tuples. So, lists are mutable, tuples are immutable.	
	To create a tuple we used parenthesis or round brackets.	

	at=(22,"n", 7.89, "now", False)	
	Same as list, we use indexing to access elements of a tuple. **at[3]** **'now'** We can't add or delete elements of a tuple, but two tuple variables can be merged into one using plus(+) operator. **bt=(34,67)** **ct=at+bt**	Remind indexing in lists Remind concatenation of two strings
	Some functions, that can be used on both list and tuple are: • **len()**: it gives the no. of elements in it. Syntax: len(list) or len(tuple) • **.count():** it gives the count of an element in a list of tuple Syntax: list.count(element)	Remind that len function gives the length of string too.
	Now, let's do an activity to revise whatever we have learned about lists and tuples.	
• **Stop sharing screen** • **Ask the student to share screen and click on Student activity 2**		
	Help the student solve all the questions. *In case the student solves all the questions too quickly, add in more questions to test use of all the functions learned till now.*	Student Activity 2: List and Tuples
• **In case the student is comfortable with Strings, List and Tuples at this point but class time remaining is less than 10 mins, just introduce the concepts of Set and Dictionary.** • **For students struggling to understand Strings, Lists and Tuples, teachers can skip Set and Dictionary.** • **If the student is able to cope up and time permits do cover both concepts and functions.**		
• **Ask the student to stop sharing screen and you share the screen**		
Concept	One more important data structure of python is **set** You might have studied in Maths, that **set is a collection of well defined and unique objects.**	
Concept	**Sets** in python are designed in a similar way. It is a collection of data values, but only unique data values, i.e. no element of a set is repeated	

	again.	
Concept	Also sets are unordered, so there is no index of any element and thus,we can't access the element of the set, individually. To create a set we simply use curly braces. **a_set={2,5.6,"toppr"}**	
Functions	We can add an element in a set using **.add()** function. Different mathematical operations that we can do on two sets are also possible in python. For any two sets, we can find their: • Union using **.union()** function • Intersection using **.intersection()**function	Teacher Activity 3: Set Operations
Concept	Another data structure in python is **Dictionary.** It is simply a collection of key-value pairs.Let's create a simple dictionary to understand it. **a_dict={1: "a",2:"b",3:"c"}** Here, 1,2,3 are the keys and a,b,c are its values respectively.	
Concept	These keys and values can be of any data type, but one thing to keep in mind is that the keys must not be repeated.	
Functions	We can use: • **.keys()** function to display all the keys of the dictionary Syntax: dictionary.keys() • **.values()** function to display all the values of the dictionary Syntax: dictionary.values() • **.items()** function to display all the key-value pairs of the dictionary Syntax: dictionary.items()	Use examples to show how to use these functions.
Functions	To access any element, **keys** can be directly used as index Syntax: dictionary[key] or we can use the **.get()** function. Syntax: dictionary.get(key)	Explain the use of both.
• **Stop sharing your screen** • **Ask student to share screen and click on Student Activity 3 only if the**		

student has covered both concept and functions		
Functions	*Help the student solve all the questions.* *In case the student solves all the questions too quickly, add in more questions to test use of all the functions learned till now.*	Student Activity 3: Sets and Dictionary
• **Ask the student to stop sharing the screen**		
	So, today we learnt some important functions related to strings, the in-built data structures of python and important functions related to them.	
	These are very important concepts, which forms the basis of a lot of advanced programs and projects.	
	So Revise all the important definitions and functions of this class and the previous one too.	
Heads up for the next class	In the next class we will be learning about different kinds of Loops	
BID GOOD BYE & END CLASS		

Resources:

Activity	Name	Links
Teacher Activity 1	Repl link	https://repl.it/languages
Teacher Activity 2	Indexing	https://drive.google.com/file/d/1fj1xAQFZXWhg8kkhGpKnqqQ75_gPPvSL/view?usp=sharing
Teacher Activity 3	Set Operations	https://drive.google.com/file/d/1KOXczUZ9z5F_kTEsT194MG740HgMEcSl/view?usp=sharing
Student Activity 1	Strings	https://repl.it/@ShailjaGupta/Strings#main.py
Student Activity 2	List and Tuples	https://repl.it/@ShailjaGupta/Lists-and-Tuples#main.py
Student Activity 3	Sets and Dictionary	https://repl.it/@ShailjaGupta/Set-and-Dictionary