While lists and strings are both sequences, a big difference between them is that lists are mutable. This means that the contents of the list can be changed, unlike strings, which are immutable. You can add, remove, or modify elements in a list.

You can add elements to the end of a list using the **append** method. You call this method on a list using dot notation, and pass in the element to be added as a parameter. For example, **list.append("New data")**would add the string "New data" to the end of the list called list.

If you want to add an element to a list in a specific position, you can use the method **insert**. The method takes two parameters: the first specifies the index in the list, and the second is the element to be added to the list.
So **list.insert(0, "New data")** would add the string "New data" to the front of the list. This wouldn't overwrite the existing element at the start of the list. It would just shift all the other elements by one. If you specify an index that’s larger than the length of the list, the element will simply be added to the end of the list.

You can remove elements from the list using the **remove** method. This method takes an element as a parameter, and removes the first occurrence of the element. If the element isn’t found in the list, you’ll get a **ValueError** error explaining that the element was not found in the list.

You can also remove elements from a list using the **pop** method. This method differs from the remove method in that it takes an index as a parameter, and returns the element that was removed. This can be useful if you don't know what the value is, but you know where it’s located. This can also be useful when you need to access the data and also want to remove it from the list.

Finally, you can change an element in a list by using indexing to overwrite the value stored at the specified index. For example, you can enter **list[0] = "Old data"**to overwrite the first element in a list with the new string "Old data".