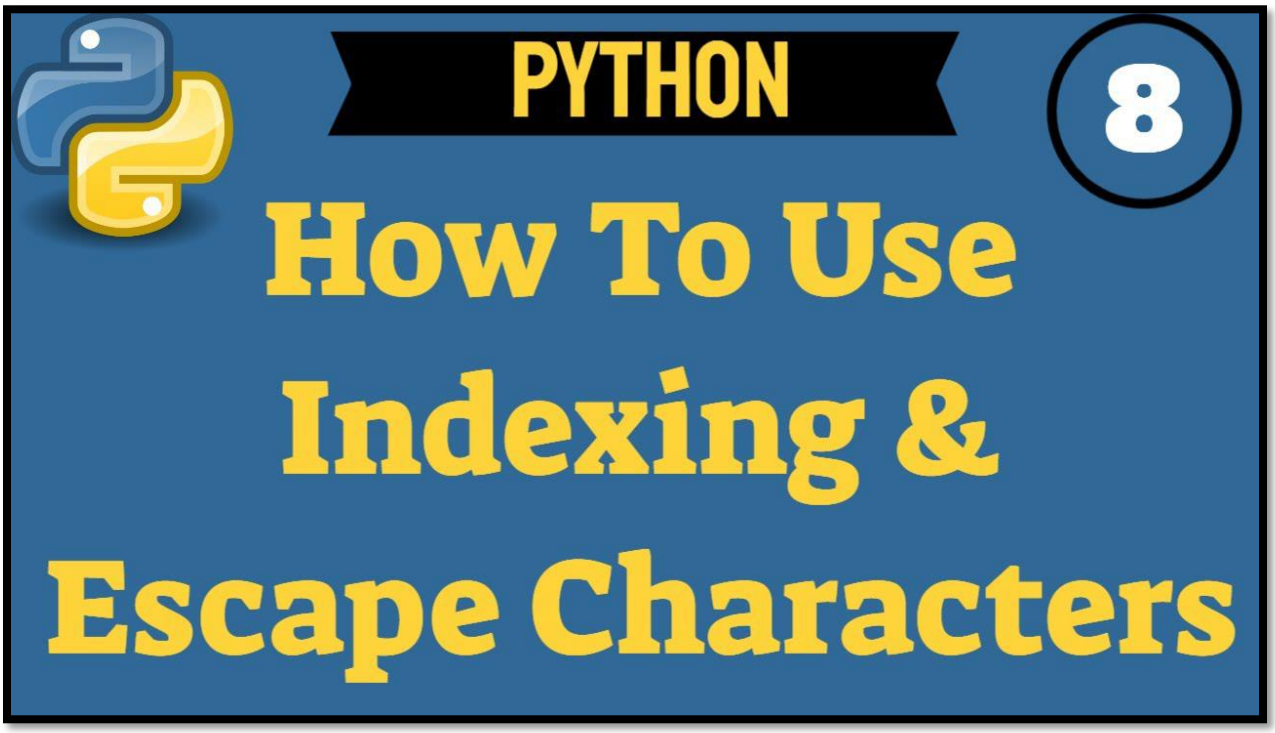
Rex
Jones II

How To Use Indexing & Escape Characters



Python Video https://www.youtube.com/watch?v=pZSWrkg85yY&list=PLfp-cJ6BH8u8iMtCoea_mWkGesJGQG-vZ&index=8

Indexing

In this tutorial session, let's talk about String Indexing and Escaped Characters. A way to think of indexing is each character in a string is associated with a number starting at zero. An escape character allows us to use characters that will not cause a syntax error.

Starting with a String index. Let's write the_truth = as a variable with a value of 'You Are Awesome'. All indexes start at zero. Therefore, the first letter which is 'Y' is index 0. I'm going to write a comment that will help us visualize the index # 0123456789. The complete string stops at index 14 including spaces.

We use the variable and square brackets to get characters from the string by writing print(the_truth[]). If we specify an index, we will get the character at that position. For example, to return the first character which is 'Y', we write 0 inside the square brackets then run.

Rex Jones II

```
the_truth = 'You Are Awesome'
#                0123456789
print(the_truth[0])
```

The Console returns Y. To return A, we change 0 to 4 because it's the 4th index. Bingo, the console returns the letter 'A'. Python also let's us use a negative index. A negative index starts from the end. Therefore we write -1 and run. The console shows 'e'. To return the 2nd capital 'A', we can either write 8 as a positive number but starting from the back -7 will also give us the letter 'A'.

```
the_truth = 'You Are Awesome'
#                0123456789
print(the_truth[-7])
```

```
A
```

Bingo, the console shows letter 'A'. Indexing a string allow us to also define a range. The range is from one index to another index. Here is the catch, it includes the starting index but does not include the ending index. The substring will get everything from the starting index then leave off the last index number. For example, the starting index is 0 and a colon (:) will help identify a range. 0 is for the letter Y and the ending index will be number 9. We see 'w' is index 9 but 'w' will not be included. The letter 'A' will be our substring last letter.

Let's Run. The Console shows You Are A.

```
the_truth = 'You Are Awesome'
#                0123456789
print(the_truth[0:9])
```

```
You Are A
```

Alright, we covered positive and negative indexes. Let me ask you a question. What will the substring be if I make the end index -1? Let's Run. You Are Awesom without an e because it's not supposed to include the last letter. I want to show you one more thing when it comes to these numbers. If a number is not included then by default it will include the last letter 'You Are Awesome'.

```
You Are Awesome
```

Escaped Characters

Now escaped characters help us do one thing. It help us to not have a syntax error. So, that's why I will copy the variable and value then show you how syntax errors are prevented using an escape character. What if I wanted to add an apostrophe to this string by writing You're. Do you see the syntax error? A syntax error is when the Python interpreter does not recognize our code. It's not valid. That's why it has a red line. The quotes let's the interpreter know when our string start and stop. In this case, the string starts at Y and stop at u. Everything after the 2nd quote is not recognized so it causes a syntax error. To fix this issue, we can replace the single quotes with double quotes.

```
the_truth = "You're Awesome"
```

Now, someone could say, I will use double quotes all the time but suppose I want to remove the apostrophe and double quotes around the word Awesome. We have another syntax error. The string You Are is included but Awesome is not included. We have a syntax error that will be fix when adding a single quote at the beginning and end. In this example, the single quotes fix the error.

```
the_truth = 'You Are "Awesome"'
```

So we see in certain cases single quotes and double quotes return a syntax error. That's why we have escaped characters and the part I wanted to show you.

Let's start over with this value and write "You Are "Awesome"". We see again the syntax error. Awesome is not recognized as part of the string. An escape character consists of a backslash followed by a character we want to add to our string. For example, we want to add double quotes around Awesome. Therefore, we ad one backslash in front of each double quote and the error goes away. To prove the error goes away, let's print(the_truth) and Run. The Console shows You Are "Awesome".

Rex Jones II

```
the_truth = "You Are \"Awesome\""
print(the_truth)
```

```
You Are "Awesome"
```

It also works for the apostrophe. Change the string value back to 'You're Awesome'. To make this error go away, we add a backslash in front of the apostrophe and run again. You're Awesome.

```
the_truth = 'You\'re Awesome'
print(the_truth)
```

```
You're Awesome
```

We can use the escape character like this backslash for a lot more commands like tab and new lines. To tab, we write You\'re Awesome \t So Smile :).

```
the_truth = 'You\'re Awesome \t So Smile :)'
print(the_truth)
```

```
You're Awesome	So Smile :)
```

Both phrases will be separated on the same line. Let's Run. You're Awesome tab So Smile. To skip a line, we change \t to \n. Now the phrases will be on different lines.

```
the_truth = 'You\'re Awesome \n So Smile :)'
print(the_truth)
```

```
You're Awesome
 So Smile :)
```

That's it for String Indexing and Escape Characters.

Rex Jones II

Contact Info

- ✔ Email Rex.Jones@Test4Success.org
- ✔ YouTube https://www.youtube.com/c/RexJonesII/videos
- ✔ Facebook https://facebook.com/JonesRexII
- ✔ Twitter https://twitter.com/RexJonesII
- ✔ GitHub https://github.com/RexJonesII/Free-Videos
- ✔ LinkedIn https://www.linkedin.com/in/rexjones34/