PYTHON PRESENTATION

Name: Siddharth Talukdar
Roll Number: 19653
Branch: ECZ
Year: 2nd (3rd Semester)

STRINGS IN PYTHON

Strings -

- Strings in Python are arrays of Unicode characters.
- However, Python does not have a character data type, a single character is simply a string with a length of 1.
- Strings in python are surrounded by either single quotation marks, or double quotation marks.
- Example: 'hello' is the same as "hello".
- A string literal can be displayed with the print() function.

Example:
print("Hello")
print('Hello')
Both the above examples will display “Hello” (Without the underlining and without the double/single Quotation marks.

Note: Quotation marks can be used inside the quotation marks as long as we make sure to give proper closure to each and every quotation mark.

Example:
print(“He is called ‘Raghav’”)
print(‘He is called “Raghav”’)
Both the above examples will display the rest of the text as it is but the First one will display ‘Raghav’, whereas the second one will display “Raghav” In the Output.

Assign String to a Variable –

Assigning a string to a variable is done with the variable name followed by an equal sign and the string.

Example: a = "Hello"

print(a)

The output of the above code will return Hello (without underlining) since it is stored in the variable named “a”.

Multiline Strings -

You can assign a multiline string to a variable by using three quotes:

Example: a = “““ Hello!

My name is Michael.

How are you today?”””

print(a)

You can use three double quotes as shown in above example which prints all the code lines as it is inside the Quotation marks.

Similarly, Three single quotes can also be used as the below example shows

Example: a = ‘’’Hello!

My name is Michael.

How are you today?’’’

print(a)

Note: In the result, the line breaks are inserted at the same position as in the code.

Strings are Arrays –

Python strings are arrays which are collection of single characters. Square brackets can be used to access elements of the string.

Example: a = "Hello, Everyone!"

print(a[1])

The output to the above example will return the element in the String at index no. 1 as “e” since indexing in arrays starts from 0 to n-1 (Where n is the number of elements in the String(Array)).

Since strings are arrays, we can loop through the characters in a string, with a for loop.

Example: for x in "banana":

print(x)

The above example will traverse through the string as it is an array and return the elements in the string after each iteration returning one element at a time in the word "banana“. This is how For loop can work in Strings.

String Length -

To get the length of a string, use the len() function. The len() function returns the length of a string

Example: a = "Hello, World!"

print(len(a))

To **check** if a certain phrase or a character belongs to a string, we use keyword in. We can also use looping like FOR loop to traverse a sentence and check if the character or word is present in a sentence. Example given below returns True.

Example: Text = “Buy one get one free!"

print("free" in Text)

Similarly To check if a word or phrase Does NOT Belong to the given String then we use keyword not in.

String Slicing –

- String Slicing is a technique that returns a subset (slice) of characters from a string using indices in arrays.
- This is done by the syntax followed for arrays. i.e. **String [start : end : interval]** where the starting point is included but the end point is not.
- If the string has “n” index and starting index is 0 then the output of slicing that string from 0 to n will return the element with index n-1 as the end point in output.
- We can use negative indexing as well to print a string in reverse by using indexing from -1 to the first element (Originally 0 index).
- The upper() function returns the string in upper case and lower() function returns string in lower case.
- We can also remove the whitespace that is the space before and/or after the actual text and very often we remove that space by using strip() function which removes the whitespace from beginning or end of the text.
- The replace() function replaces a string with another string. The split() splits the string into substrings when we provide the separator.

String Concatenation –

To concatenate, or combine, two strings you can use the + operator.

Example:

```
a = “Hello”
b = “World”
c = a + b
Print(c)
```

The output of the above code will return the output as HelloWorld.