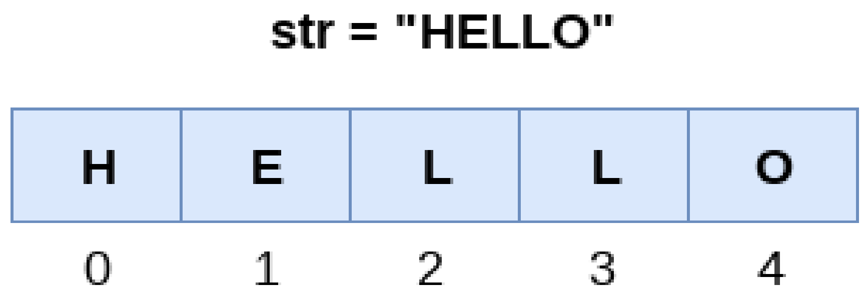
Strings

- Strings in python are surrounded by either single quotation marks, or double quotation marks.
- 'hello' is the same as "hello".
- a = "Hello"
  print(a)

```
#Using single quotes
str1 = 'Hello Python'
print(str1)
#Using double quotes
str2 = "Hello Python"
print(str2)

#Using triple quotes
str3 = '''Triple quotes are generally used for
    represent the multiline or
    docstring'''
print(str3)
```

Strings indexing
Strings are Arrays



str[0] = 'H'

str[1] = 'E'

str[2] = 'L'

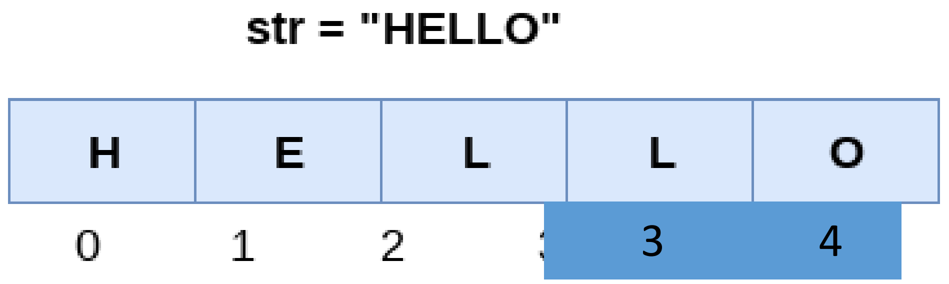
str[3] = 'L'

str[4] = 'O'

```
str = "HELLO"
print(str[0])
print(str[1])
print(str[2])
print(str[3])
print(str[4])
# It returns the IndexError because 6th index doesn't exist
print(str[6])
```

Slicing

use the : (colon) operator in Python to access the substring from the given string.



str[0] = 'H'

str[1] = 'E'

str[2] = 'L'

str[3] = 'L'

str[4] = 'O'

str[:] = 'HELLO'

str[0:] = 'HELLO'

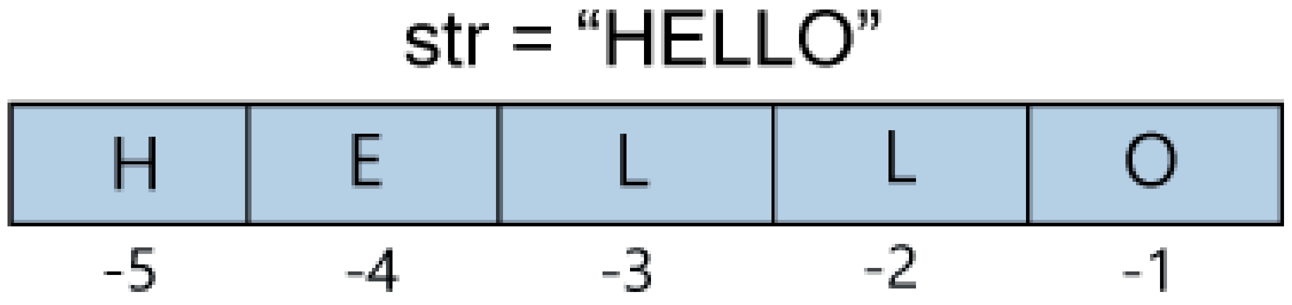
str[:5] = 'HELLO'

str[:3] = 'HEL'

str[0:2] = 'HE'

str[1:4] = 'ELL'

- We can do the negative slicing in the string; it starts from the rightmost character, which is indicated as -1. The second rightmost index indicates -2, and so on



```
str[-1] = ‘O’        str[-3:-1] = ‘LL’
str[-2] = ‘L’        str[-4:-1] = ‘ELL’
str[-3] = ‘L’        str[-5:-3] = ‘HE’
str[-4] = ‘E’        str[-4:] = ‘ELLO’
str[-5] = ‘H’        str[::-1] = ‘OLLEH’
```

String Length

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

```
a = "Hello, World!"
print(len(a))
```

Looping Through a String

```
for x in "banana":
  print(x)
```

Check String: to check phrase or character is present in a string, we can use the keyword **in**

```
txt = "The best things in life are free!"
print("free" in txt)
```

- The upper() method returns the string in upper case:

```
a = "Hello, World!"
print(a.upper())
```

- The lower() method returns the string in lower case:

```
a = "Hello, World!"
print(a.lower())
```

- The strip() method removes any whitespace from the beginning or the end:

```
a = "      Hello, World!      "
print(a.strip())
 # returns "Hello, World!"
```

- The replace() method replaces a string with another string:

```
a = "Hello, World!"
print(a.replace("H", "J"))
```

- The split() method splits the string into substrings if it finds instances of the separator:

```
a = "Hello, World!"
print(a.split(“,")) # returns ['Hello', ' World!']
```

String Concatenation

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

```
a = "Hello"
b = "World"
c = a + b
print(c)
```

```
str = "Hello"
str1 = " world"
print(str*3) # prints HelloHelloHello
print(str+str1)# prints Hello world
print(str[4]) # prints o
print(str[2:4]); # prints ll
print('w' in str) # prints false as w is not present in str
print('wo' not in str1) # prints false as wo is present in str1.
print("The string str : %s"%(str)) # prints The string str : Hello
```

Python String Formatting
Escape Sequence

```
str = "They said, "Hello what's going on?""
print(str)
```

Output:

```
SyntaxError: invalid syntax
```

We can use the triple quotes to accomplish this problem but Python provides the escape sequence.

The backslash(/) symbol denotes the escape sequence

```
# escaping single quotes
print('They said, "What\'s going on?"')

# escaping double quotes
print("They said, \"What's going on?\"")
```

Homework

1. Make a list of an escape sequence and write example for each
2. Read about more string methods(built-in methods), u can refer this:

https://www.w3schools.com/python/python_ref_string.asp

```
str = 'Helloworld'
print(str[-1])
print(str[-3])
print(str[-2:])
print(str[-4:-1])
print(str[-7:-2])
# Reversing the given string
print(str[::-1])
print(str[-12])
```

The format() method:The **format()** method is the most flexible and useful method in formatting strings. The curly braces {} are used as the placeholder in the string and replaced by the **format()** method argument.

# Using Curly braces

print("{} and {} both are the best friend".format("Devansh","Abhishek"))

 #output: Devansh and Abhishek both are the best friend

#Positional Argument

print("{1} and {0} best players ".format("Virat","Rohit"))

#Keyword Argument

print("{a},{b},{c}".format(a = "James", b = "Peter", c = "Ricky"))

Output:

Rohit and Virat best players

James,Peter,Ricky

```
I = 10
F = 1.290
S = "Devansh"
print("Hi I am Integer ... My value is %d\nHi I am float ... My value is %f\nHi I am string ... My value is %s"%(I,F,S))
```

Output:

```
Hi I am Integer ... My value is 10
Hi I am float ... My value is 1.290000
Hi I am string ... My value is Devansh
```

```
print("Hi I am Integer ... My value is %d\nHi I am float ... My value is %.10f\nHi I am string ... My value is %s"%(I,F,S))
Hi I am Integer ... My value is 10
Hi I am float ... My value is 1.2000000000
Hi I am string ... My value is Devansh
```

We previously use the str.format() method mostly to format the strings. But, the time has changed we have a new method to make our efforts twice as fast.

The variables in the curly { } braces are displayed in the output as a normal print statement. Let's see an example.

```
## declaring variables
name = "Datacamp"
type_of_company = "Educational"

## enclose your variable within the {} to display it's value in the output
print(f"{name} is an {type_of_company} company.")
```

String Operators

Operator	Description
+	It is known as concatenation operator used to join the strings given either side of the operator.
*	It is known as repetition operator. It concatenates the multiple copies of the same string.
[]	It is known as slice operator. It is used to access the sub-strings of a particular string.
[:]	It is known as range slice operator. It is used to access the characters from the specified range.
in	It is known as membership operator. It returns if a particular sub-string is present in the specified strin
not in	It is also a membership operator and does the exact reverse of in. It returns true if a particular s present in the specified string.
r/R	It is used to specify the raw string. Raw strings are used in the cases where we need to print the act escape characters such as "C://python". To define any string as a raw string, the character r or R is f string.
%	It is used to perform string formatting. It makes use of the format specifiers used in C programming l map their values in python. We will discuss how formatting is done in python.

Thank You