Slicing in Python

Slicing allows you to extract a portion (substring or sublist) of a sequence (like a string, list, or tuple).

Syntax:

```
sequence[start:stop]
```

- `start`: index to begin slicing (inclusive)
- `stop`: index to end slicing (exclusive)

Example:

```
s = "Python"
print(s[0:4])  # Output: Pyth

str1=“hello world”
#accessing a substring
print(str1[1:5])

#updating a string
print(str1[:6] + 'Python‘)
```

Step Slicing in Python

Adds a step value to the slice, allowing you to skip elements.

Syntax:

```
sequence[start:stop:step]
```

Example:

```
s = "Python"
print(s[0:6:2])  # Output: Pto
```

Backward Slicing in Python

Used to reverse a sequence using negative step.

Example:

```
s = "Python"
print(s[::-1])  # Output: nohtyP
```

String Operators in Python

Concatenation (+):

```
"Hello " + "World"  # Output: Hello World
```

Repetition (*):

```
"Hi! " * 3  # Output: Hi! Hi! Hi!
```

Membership (in, not in):

```
"P" in "Python"  # Output: True
"P" not in "Python"  # Output: False
```

String Replacement Fields

Used with `.format()` or f-strings to insert values into strings.

Example:

```
"Hello, {}".format("Alice")  # Output: Hello, Alice
```

Or with index:

```
"{0} is {1} years old".format("Bob", 25)
```

How to Format Strings

You can format strings in several ways:

`str.format()` method:

```
print("My name is {} and I am {} years old.".format("John", 30))
```

`%` formatting (older style):

```
Print("My name is %s and I am %d years old." % ("John", 30))
```

f-strings (Python 3.6+):

```
name = "John"
age = 30
print(f"My name is {name} and I am {age} years old.")
```

Precision in Python (for strings and numbers)

You can format numbers with a specific precision using format specifiers:

Example:

```
pi = 3.14159265
print("{:.2f}".format(pi))  # Output: 3.14

# Or with f-strings:
print(f"{pi:.3f}")  # Output: 3.142
```

---

f-strings in Python

Formatted string literals introduced in Python 3.6.

Syntax:

```
name = "Alice"
age = 28
print(f"My name is {name} and I am {age} years old.")
```

- You can embed expressions:

```
print(f"{2 + 3}")  # Output: 5
```

String Interpolation

Refers to inserting variables into strings. Python supports several ways:

f-strings (modern, preferred):

```
print(f"Hello {name}")
```

`str.format()`:

```
print("Hello {}".format(name))
```

`%` formatting (legacy):

```
print("Hello %s" % name)
```

```
import datetime
today = datetime.datetime.today()
print(f"{today:%B %d, %Y}")
print(f"{today:%b %d, %Y,%H:%M:%S}")
```

White space Characters:-

\n Newline
\t Tab

Example:-

```
Doc_str = "my name is srinivas\tand  I am student\n, and learning python"
print (Doc_str)
```

Sting functions:-

```
str1 = “piyush is python professional!!!"
str2 = “python"
print (str1.find(str2)) # 9th position
print (str1.find(str2, 9))
print (str1.find(str2, 10)) # it will return -1 if not found substring
print (str1.index(str2))
print (str1.index(str2, 9))
print (str1.index(str2, 10)) # it will return error if not found substring
print (str1.rfind(str2))
print (str1.rindex(str2))

str1 = “/“
 st = (“c:", “documents", “srinivas")
print (str1.join( st ))

str1=“   abc    “
print(len(str1))
print(str1.upper())
print(str1.lower())
print(str1.lstrip())
#Removes all leading whitespace in string
print(str1.rstrip())
print(str1.strip())
print(max(str1))
print(min(str1))
str2=“     “
print(str2.isspace())

txt = "welcome to the uae“
x = txt.split()
print(x)
txt = "Thank you for the music\nWelcome to the uae“
x = txt.splitlines()
print(x)
str1 = “salim is python professional!!!"
substr = “python"
print ("sub string count", str1.count(substr))
```

Numbers:-

```
V=10
del V
x=2
print(int(x))
print(float(x))
print(complex(x)) #2+0j
y=5
print(complex(x, y))#2+5j
```

Mathematical constants:-

```
>>>import math
>>> math.e
2.718281828459045
>>> math.pi
3.141592653589793
```

Mathematical functions:-

```
import math
print(math.sqrt(9))
```

- math.ceil(*x*)-Return the ceiling of *x*, the smallest integer greater than or equal to *x*.
- math.fabs(*x*)Return the absolute value of *x*.
- math.factorial(*x*)Return *x* factorial.
- math.floor(*x*)Return the floor of *x*, the largest integer less than or equal to *x*.
- math.exp(x) The exponential of x: e^xf
- math.log(x)The natural logarithm of x, for x> 0
- math.log10(x)The base-10 logarithm of x for x> 0 .
- max(x1, x2,...)The largest of its arguments: the value closest to positive infinity
- min(x1, x2,...) The smallest of its arguments: the value closest to negative infinity
- math.pow(x, y)The value of x**y.
- round(x,n)**x** rounded to n digits from the decimal point. Python rounds away from zero as a tie-breaker: round(0.5) is 1.0 and round(-0.5) is -1.0.
- math.sqrt(x)The square root of x for x > 0

Random Modules:-

```
import  random
print(random.choice([1, 2, 3, 5, 9]))
print(random.randrange(100, 1000))
print(random.uniform(5, 10) )
#Return random number between 5.0 to 9.99
print ("random() : ", random.random())
#Return random number between 0.0 to 0.99
list1=[1,2,3,4]
random.shuffle(list1)
print(list1)
random.shuffle(list1)
print(list1)
```

OS Modules:-

```
import os
print(os.getcwd())
os.mkdir("newdir")
os.rmdir('newdir')
os.chdir(r"c:/users/a")
print(os.getcwd())
os.mkdir("newdir")
os.remove("file1.txt")
```