Working with Python Strings

North Austin Pythonistas
September 2019

Goals

* String Literals
* Comparing Strings
* Finding Sub-Strings
* Parsing Strings
* Constructing Strings
* Translating Characters in a String

String Literals

* String Literal - **Interpolates Escape Codes**

  “\ta string” -> “	a string”

* Raw String Literals - No Interpolation

  r”\tfoo” != “\tfoo”

* Byte Literals - bytes() class

  b”this is a byte buffer”

Comparing Strings

* (“foo” == “foo”) == True
* “oba” **in** “foobar” == True
* “foobar”.**startswith**(“foo”) == True
* “foobar”.**endswith**(“bar”) == True

Finding Substrings

* “foobar”.**find**(“ob”) == 2

  Returns -1 if substring not found.

* “foobar”.**index**(“o”) == 1

  Raises ValueError if substring not found.

* “foobar”.**count**(“o”) == 2

Parsing Strings

* “foo#bar”.**split**(‘#’) == [“foo”, “bar”]
* A, B, C = “baz#ack”.**partition**(“#”)

  A == “baz”, B == “#”, C == “ack”
* lines = “D\nE\nF\n”.**splitlines**()

  lines == [“D”, “E”, “F”]

Constructing Strings

* “-“.join([“foo”, “bar”])
* “foo” + “-” + “bar” # looks like Java, yuck!
* “name: %s age: %d” % (“erik”, “18”)
* “name: {} age: {}”.format(name, age)
* f”name: {name} age: {age}” # python ^3.6

Constructing Strings – continued

* “foobar”.**replace**(“bar”, “ack”) == “fooack”

Translating Characters in a String

```
>>table = str.maketrans( {“f”:”g”, “b”:”t”} )

>>“foobar”.translate(table) == “gootar”
```

Bonus Topic! Python3’s Enhanced Print

```
print(value, …, sep=“ ”, end=“\n”,
        file=sys.stdout, flush=False)
```

* `print(“literal”, name) -> “literal erik”`
* `print(*[“foo”, “bar”], sep=“#”) -> “foo#bar”`
* `print(line, file=open(“text.out”, “w”))`