Diving In

In Python 3, all strings are sequences of Unicode characters. There is no such thing as a Python string encoded in `UTF-8`, or a Python string encoded as CP-1252. “Is this string `UTF-8`?” is an invalid question. `UTF-8` is a way of encoding characters as a sequence of bytes. If you want to take a string and turn it into a sequence of bytes in a particular character encoding, Python 3 can help you with that. If you want to take a sequence of bytes and turn it into a string, Python 3 can help you with that too. Bytes are not characters; bytes are bytes. Characters are an abstraction. A string is a sequence of those abstractions.

```
s = '&^ Python'              #①
print (len(s))               #②
#9

print (s[0])                 #③
#&

print (s + ' 3')             #④
#&^ Python 3
```

① To create a string, enclose it in quotes. Python strings can be defined with either single quotes (’) or double quotes (").

② The built-in `len()` function returns the length of the string, i.e. the number of characters. This is the same function you use to find the length of a list, tuple, set, or dictionary. A string is like a tuple of characters.

③ Just like getting individual items out of a list, you can get individual characters out of a string using index notation.

④ Just like lists, you can concatenate strings using the `+` operator.