Tuple Slicing

This notebook discusses slicing as it applies to tuples. Like in the case of strings, we can use the slicing feature of Python to extract or otherwise manipulate portions of a tuple.

`[start_val:end_val:step_val]`

The slice begins with the element whose index is given by `start_val` (default 0), until 1 less than the `end_val`, in increments of the `step_val` (default is 1). If the `end_val` is not provided, the default is until the end of the tuple.

In [1]:

```
my_tuple = (1, 2, 3, 4, 5, 6, 7, 8)
print('my_tuple[2:4]', my_tuple[2:4])
print('my_tuple[5:]', my_tuple[5:])
print('my_tuple[:5]', my_tuple[:5])
print('my_tuple[:]', my_tuple[:])
print('my_tuple[-1:-8:-1]', my_tuple[-1:-8:1])
print('my_tuple[-1:-9:-1]', my_tuple[-1:-9:-1])
```

```
my_tuple[2:4] (3, 4)
my_tuple[5:] (6, 7, 8)
my_tuple[:5] (1, 2, 3, 4, 5)
my_tuple[:] (1, 2, 3, 4, 5, 6, 7, 8)
my_tuple[-1:-8:-1] ()
my_tuple[-1:-9:-1] (8, 7, 6, 5, 4, 3, 2, 1)
```

Note that since tuples are immutable, you cannot delete portions of a tuple.

In [2]:

```
elements = ("A", "B", "C", "D")
del elements[:1] #will generate an error
print(elements)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-2-060fdef77834> in <module>
      1 elements = ("A", "B", "C", "D")
----> 2 del elements[:1] #will generate an error
      3 print(elements)

TypeError: 'tuple' object does not support item deletion
```