Ch07-Tuples

September 10, 2025

1 Tuples

- http://openbookproject.net/thinkcs/python/english3e/tuples.html
- containers used for grouping data values surrounded with parenthesis
- data values in tuples are called elements/items/members
- two major operations done with tuples are:
  1. packing (creating tuples)
  2. unpacking (storing data into individual variables)

[1]:
```
year_born = ("Paris Hilton", 1981) # tuple packing
```

[2]:
```
print(year_born)
```

```
('Paris Hilton', 1981)
```

[11]:
```
star = "Paris", 'J', 'Hilton', 1981, 32, 1.2 # tuple packing without parenthesis
```

[12]:
```
star
```

[12]:
```
('Paris', 'J', 'Hilton', 1981, 32, 1.2)
```

[13]:
```
type(star)
```

[13]:
```
tuple
```

[14]:
```
# tuple assignment
fname, mi, lname, year, age, income = star # tuple unpacking
# no. of variables must match no. values in tuple
```

[9]:
```
fname
```

[9]:
```
'Paris'
```

[10]:
```
lname
```

[10]:
```
'Hilton'
```

[11]:
```
print(income)
```

```
1.2
```

[12]:

```
# swap values of two variables
a = 100
b = 200
a, b = b, a
```

[13]:

```
print(a, b)
```

```
200 100
```

1.1 Member access

- each member of tuple can be accessed using `[ index ]` operator
- index is 0-based or starts from 0

[1]:

```
name = ('John', 'A.', 'Smith')
```

[3]:

```
print(name[0], name[1], name[2])
```

```
John A. Smith
```

1.2 Length of tuple

- `len()` gives the length (no. of elements) of tuple

[4]:

```
len(name)
```

[4]:

```
3
```

1.3 Tuple membership

- **in** and **not in** boolean operators let's you check for membership

[7]:

```
'John' in name
```

[7]:

```
True
```

[8]:

```
'B.' in name
```

[8]:

```
False
```

[9]:

```
'Jake' not in name
```

[9]:

```
True
```

1.4 Function can return multiple values as Tuple

- multiple comma separated values can be packed and returned as tuple from function

[2]:

```
def maxAndMin(a, b, c, d, e):
    myMax = a #max(a, b, c, d, e)
    if myMax < b:
        myMax = b
    if myMax < c:
        myMax = c
    if myMax < d:
        myMax = d
    if myMax < e:
        myMax = e
    values = [a, b, c, d, e]
    myMin = min(values)
    return myMax, myMin
```

[4]:

```
ab = maxAndMin(10, 20, 5, 100, 34)
print(f'max = {ab[0]} and min = {ab[1]}')
```

```
max = 100 and min = 5
```

1.5 Tuples are immutable

- can't change tuple in-place or update its elements
  - similar to string

[5]:

```
a = (1, 2, 3)
print(a[0])
```

```
1
```

[6]:

```
a[0] = 100
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-6-b6d7a4db9a51> in <module>()
----> 1 a[0] = 100

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

1.6 Applications of Tuple

- application of tuples is limited due to its immutability in nature

[ ]: