Programming Fundamentals

Data Types: Tuples

João Correia Lopes

INESC TEC, FEUP

06 November 2018

GOALS

By the end of this class, the student should be able to:

- Describe how to work with tuples
- Enumerate the main methods available to work with tuples

BIBLIOGRAPHY

- Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers, How to Think Like a Computer Scientist — Learning with Python 3, 2018 (Section 5.2) [PDF]
- Brad Miller and David Ranum, Learning with Python: Interactive Edition. Based on material by Jeffrey Elkner, Allen B. Downey, and Chris Meyers (Section 10.26, Section 10.27, Section 10.28) [HTML]
- Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers, How to Think Like a Computer Scientist — Learning with Python 3 (RLE), 2012 (Chapter 9) [HTML]

TIPS

- There's no slides: we use a script and some illustrations in the class. That is NOT a replacement for **reading the bibliography** listed in the *class plan*
- "Students are responsible for anything that transpires during a class—therefore **if you're not in a class**, you should get notes from someone else (not the instructor)"—David Mayer
- The best thing to do is to **read carefully** and **understand** the documentation published in the Content wiki (or else **ask** in the class)
- We will be using **Moodle** as the primary means of communication

Contents

A COMPOUND DATA TYPE

- So far we have seen built-in types like `int`, `float`, `bool`, `str` and we've seen lists and pairs
- Strings, lists, and **tuples** are qualitatively different from the others because they are made up of smaller pieces
- Tuples group any number of items, of different types, into a single compound value
- Types that comprise smaller pieces are called **collection** or **compound data types**
- Depending on what we are doing, we may want to treat a compound data type as a single thing

TUPLES ARE USED FOR GROUPING DATA

- A **data structure** is a mechanism for grouping and organizing data to make it easier to use
- We saw earlier that we could group together pairs of values:

```
>>> year_born = ("Paris Hilton", 1981)
```

- The pair is an example of a **tuple**
- Generalizing this, a tuple can be used to group any number of items into a single compound value

```
>>> julia = ("Julia", "Roberts", 1967, "Duplicity", 2009, "Actress", "
    Atlanta, Georgia")
>>>
>>> empty_tuple=()
```

OPERATIONS ON TUPLES

- A tuple lets us “chunk” together related information and use it as a single thing
- Tuples support the same sequence operations as strings
- The index operator selects an element from a tuple

⇒ `https://github.com/fpro-admin/lectures/blob/master/11/tmethods.py`

TUPLE ASSIGNMENT

- Python has a very powerful tuple assignment feature
- Allows a tuple of variables on the left of an assignment to be assigned values from a tuple on the right of the assignment

```
 (name, surname, year_born, movie, year_movie, profession, birthplace) =
     julia
```

⇒ https://github.com/fpro-admin/lectures/blob/master/11/tassign.py

Tuples as return values

- Functions can always only return a single value
- by making that value a tuple, we can effectively group together as many values as we like, and return them together

⇒ https://github.com/fpro-admin/lectures/blob/master/11/circle_stats.py

COMPOSABILITY OF DATA STRUCTURES

- Tuples items can themselves be other tuples
- Tuples may be **heterogeneous**, meaning that they can be composed of elements of different type

```
julia_more_info = ( ("Julia", "Roberts"), (8, "October", 1967),
                     "Actress", ("Atlanta", "Georgia"),
                     [ ("Duplicity", 2009),
                       ("Notting Hill", 1999),
                       ("Pretty Woman", 1990),
                       ("Erin Brockovich", 2000),
                       ("Eat Pray Love", 2010),
                       ("Mona Lisa Smile", 2003),
                       ("Oceans Twelve", 2004) ])
```

UPDATING TUPLES

- Tuples are immutable, which means you cannot update or change the values of tuple elements
- You are able to take portions of existing tuples to create new tuples

```python
tup1 = (12, 34.56)

# Following action is not valid for tuples
# tup1[0] = 100

# So let's create a new tuple as follows
tup1 = (100,) + tup1[1:]
print(tup1)
```

⇒ https://github.com/fpro-admin/lectures/blob/master/11/tmore.py

EXERCISES

- Moodle activity at: LE11: Tuples