Collections and Casting in Python

Data Structures

Lecturer: Caio Fonseca

Introduction

- This section will briefly introduce the Python Collections and Casting in Python
- The purpose is to make you learn about the different use and characteristics of each Python collection types.

What is a Collection?

- A collection is simply an object that allows you to store multiple items in a single place (object).
  - Q. Can you think of any examples?
  - Q. What is the advantage of having a single storage place?
- Methods or facilities will be provided to access and manipulate what is stored in a collection.
  - Add, retrieve, remove, aggregate, etc.
- Collections tend to store groups of related items, that is, items that logically belong together.
  - Q. Can you think of any examples?
  - Q. Does this grouping remind you of any other aspects of computing you have studied?

What Can We Store in a Collection?

- Typically, we can store anything in a collection.
- Collections may be typed
  - For instance, arrays are typed (e.g. integer arrays) in many programming languages.
  - It is good practice to specify the types of objects being stored in collections where possible to avoid errors.
- Many variants of collections:
  - Some collections allow duplicates; other don't. Some keep elements sorted; others don't, etc.

Collections

- Array
  - Collection of elements of one type (e.g. integers). Can be multidimensional.
- Vector
  - Growable/shrinkable array.
- Hashtable
  - Maps keys to values.

Descriptions of the Core Collections

- Collection
  - Generic root, which all others implement.
- Set
  - No duplicates. Mathematical set. No order.
  - No extra methods over Collection. Simply restricts duplicates.
- List
  - Ordered or sequenced. Duplicates permitted.

Continued…

- Queue
  - Ordered items (e.g. FIFO). Extra manipulations for insertion, inspection, extraction, etc.
- Map
  - Maps (i.e. associates one-to-one) keys to values.
- SortedSet/SortedMap
  - Variants that keep elements sorted in assending order. They also add facilities to take advantage of this.

Collections in Python

There are four collection data types in Python:

- List
  - Ordered items, changeable. Allow duplicates;
- Tuple
  - Ordered collection, unchangeable. Also allow for duplicate members;
- Set
  - Unordered and unindexed. No duplicate members.
- Dictionary
  - Unordered and changeable. No duplicate members.

https://www.w3schools.com/python

Casting

- Casting allows us to convert an object reference of one type into an object reference of another type in order to use it as that alternative type.
- Both “up casting” and “down casting” can be used.
- Generally speaking, up casting is largely implicit and automatic. For instance, invoking getName() on a Student object would implicitly up cast the object to type Person where the getName() method generically resides. Similarly, a Student object can always be treated as a Person or Object (e.g. passed to methods or stored as references) through implicit up casting.
- Down casting is more explicit as it requires the programmer to state what sub-type an object should be cast down to. The object type cast is specified in round brackets before the object reference to do this.

Casting in Python

- Casting in Python can be done easily with constructor built-in functions:
  - Int() – Will allow to construct an integer number from an integer literal, float literal (rounding down to the whole number) or a string literal (given that the string represents a whole number);
  - Float() – Allows the construction of a float number from an integer literal, a float literal or a string literal (if the string represents a float or an integer);
  - Str() – Will allow the construction of a string from a wide variety of data types (float, integers and string literals).

https://www.w3schools.com/python/python_casting.asp