

6.100L Recitation 5 – 14 October 2022

Reminders:

- MQ5 Monday 10/17
- PS 2 due Wednesday 10/19
- Remember to complete checkoff!

Lecture 9 & 10 Recap – Lists, Tuples & Mutability

1. Tuples

- These are ordered sequences of objects. These objects can be of any type.
- immutable, i.e cannot be changed once created
- can be indexed
- Iterable - can loop over them
- you can slice a tuple giving you a subset of the original tuple

tuple1 = (1, 2, 3, 4)

len(tuple1) # gives you the length of the tuple

tuple1[0:2] # gives (1,2)

- Functions often return tuples as a way of returning multiple values

Return a, b # last line of the function where a, b is the tuple (a, b)

1. Lists

- ordered sequence of objects
- can be indexed & sliced similarly to tuples
- Iterable - can loop over them
- mutable, i.e. can be changed/modified after being created
 - For example given the two lists:

list1 = [1,2,3, “MIT”]

list2 = [4,5,6]

- You can change the element at index 0 with

list1[0] = 5

- add an element to the end

list1.append(5)

- add all elements of list2 to the end of list1 with

list1.extend(list2)

- remove an element at specific index with

del list1[index]

- remove element at the end

list1.pop()

- remove a specific element with

```
list1.remove("MIT")
```

- note that if an element appears multiple times, this method will only remove the first occurrence of that element
- if the element is not present, throws error

Useful Methods

- `my_list.copy()` # no mutation - returns copy
- `my_list.reverse()` # mutation
- `sorted(my_list)` # no mutation – returns sorted list
- `my_list.sort()` # mutation
- `my_list.extend([x,y])` # mutation
- `my_list[:]` # makes clone
- `my_list.remove(2)` # removes the first occurrence of 2 in the list
- `my_list.pop()` # pops last element - mutation
- `my_list.pop(2)` # pops 3rd element
- `my_list.insert(1, 7)` # inserts 7 in the 2nd position - mutation

Immutable vs Mutable Data structures

- **Immutable data types:** cannot change element value after assignment
 - Examples of immutable data types we've seen:
 - int
 - float
 - bool
 - string
 - tuple
- **Mutable data types:** can change element after assignment
 - We can think of mutable objects as being assigned to a certain place in memory. In this case, assigning a variable to a mutable object just means that it points to that object in memory.
 - Multiple variables can point to the same object in memory. This can be problematic because mutating a variable will affect the other variables that point to it. This is called aliasing.
 - Examples of mutable data types we've seen:
 - lists
 - Dictionaries (have not seen)

MIT OpenCourseWare

<https://ocw.mit.edu>

6.100L Introduction to CS and Programming Using Python

Fall 2022

For information about citing these materials or our Terms of Use, visit: <https://ocw.mit.edu/terms>