Data Structures: Lists, Tuples & Dictionaries

Computing for Data Analytics (CPSC 4800)

Mourad Bouguerra
mourad.bouguerra@me.com

Langara College

Summer 2023

Lesson's Outline

Learning Objectives

Learning Objectives

✎ Upon completion of this lesson, you will learn:

- ❒ Python key data structures
  - ➡ Python lists
  - ➡ Python tuples
  - ➡ Python dictionaries

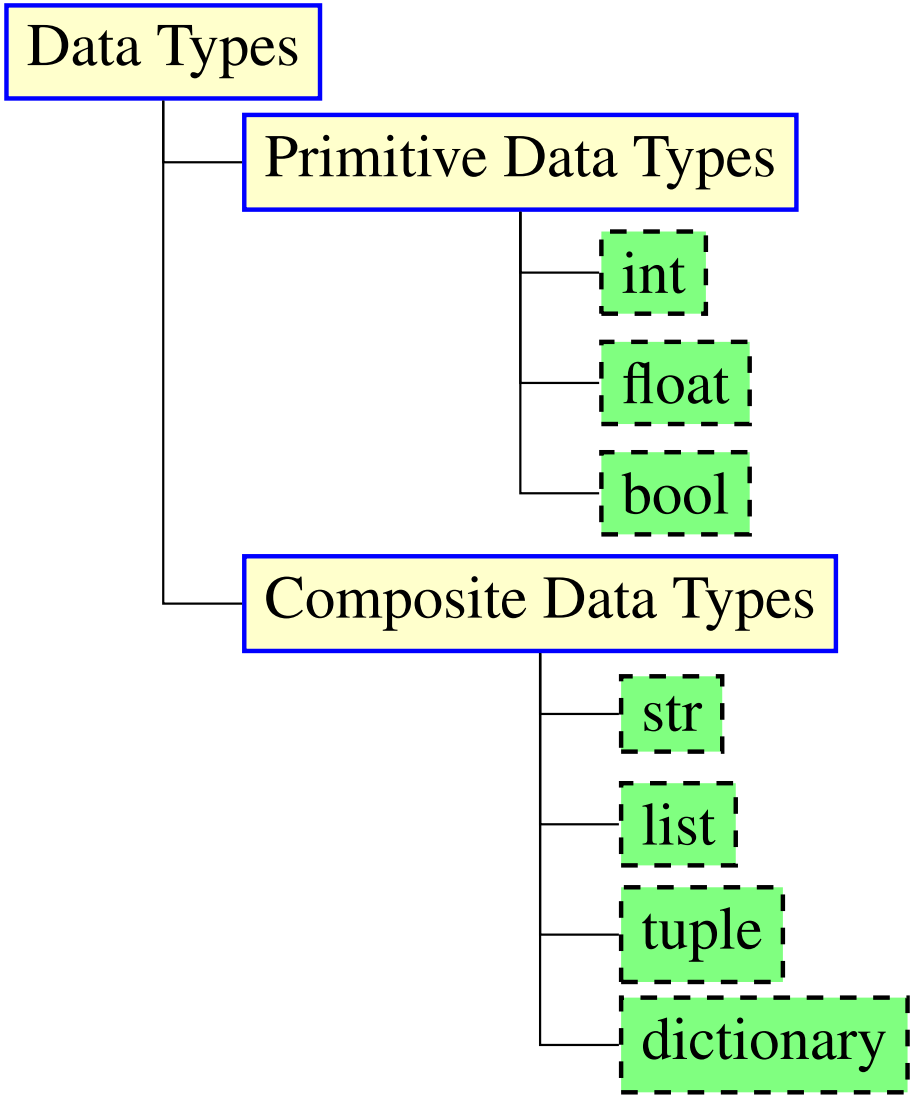
Python Built-in Data Types

Category Type	Identifiers	Example
Text/String	`str`	`pep_8_url = 'https://peps.python.org/pep-0008/'`
Numeric	`int, float, complex`	`course_credit = 3`
Boolean	`bool`	`is_transferable = True`
Sequence	`list,tuple,range`	`vowels=['a','e','i','o','u']`
Mapping	`dict`	`course={'code':'CPSC 4800','credit':3}`
Set	`set`	`my_set={3,'CPSC 4800', True}`

Python Built-in Data Types

Category Type	Identifiers	Example
Text/String	`str`	`pep_8_url = str('https://peps.python.org/pep-0008/'`
Numeric	`int, float, complex`	`course_credit = int(3)`
Boolean	`bool`	`is_transferable = bool(True)`
Sequence	`list,tuple,range`	`vowels=list('a','e','i','o','u')`
Mapping	`dict`	`course=dict(code='CPSC 4800',credit=3)`
Set	`set`	`my_set=set(3,'CPSC 4800', True)`

Python Built-in Data Types



Python List Data Structure

Lists

- A list is a key Python data structure that stores
  - a collection of data as an ordered sequence
    - in a continuous memory locations
- A list can hold different data types
- To get the number of elements use the method *len*()

```
course_info = ['Computing for Data Analytics', 'CPSC 4800',3,\
               True,False,32.40]
print(f'The list {course_info} has {len(course_info)} elements.')
```

Accessing List Elements

- Elements of the list are accessed using positive indices
  - 0 index to access the first element `course_info[0]`
  - 1 index to access the second element `course_info[1]`
  - 2 index to access the third element `course_info[2]`
  - .....

```
print f'The first element in the list is {course_info[0]}.'
print f'The second element in the list is {course_info[1]}.'
print f'The third element in the list is {course_info[2]}.'
```

Accessing List Elements

- Elements of the list are accessed using negative indices
  - -1 index to access the last element `course_info[-1]`
  - -2 index to access the second last element `course_info[-2]`
  - -3 index to access the third last element `course_info[-3]`
  - .....

```
print f'The last element in the list is {course_info[-1]}.'
print f'The second last element in the list is {course_info[-2]}.'
print f'The third last element in the list is {course_info[-3]}.'
```

Accessing List Elements

Positive Indices		**0** ↓	**1** ↓	**2** ↓	**3** ↓	**4** ↓	**5** ↓	
List	[	9,	$'Python'$,	$True$,	$1e6$,	$(7, 13)$,	$\{'key' :'$**4800**$'\}$	]
Nwgative Indices		↑ -6	↑ -5	↑ **-4**	↑ **-3**	↑ **-2**	↑ **-1**	

List Content

- A list is a mutable object
  - its content can be changed

```
course_info    =
course_info -   =
print f'The list after making changes is\n{course_info}.'
# Output
# The list after making changes is
# ['Computing for Data Analytics', 'CPSC 4800', 4, True, False, 50].
```

Class Activity

- Giving the following Python list

```
vancouver = 'Vancouver'                    -        False  'Surrey' 'Burnaby'
```

- Use positive indices to access each element of the list and display its type

- Use negative indices to access each element of the list and display its type

Chinese Proverb

Tell Me & I Forget,
Teach Me & I Remember,
Involve Me & I Learn

List Slicing

- List slicing allows to acces a sublist of the given list
- To slice a list my_list
  - $my\_list[start : end]$ returns the sublist
    - ✔ elements with indices from $start$ to $end - 1$

```
course_info =  'Computing for Data Analytics'  'CPSC 4800'
                True  False
print f'{course_info[2:4]}'
# output
# [3, True]
```

List Slicing

- List slicing allows to acces a sublist of the given list
- To slice a list my_list
  - $my\_list[start : end : step]$ returns the sublist
    - ✔ elements with indices from $start$ to $end - 1$ using a step

```
course_info =  'Computing for Data Analytics'  'CPSC 4800'
                True  False
print f'{course_info[1:4:2]}'
# output
# ['CPSC 4800', True]
```

List Slicing

Example	Description
`my_list[:]`	Returns all the elements of the list
`my_list[start:]`	Returns all elements from start index to the end
`my_list[:end]`	Returns all elements from the beginning to the element with index $end - 1$
`my_list[-1::-1]`	Returns the reversed list

Class Activity

- Giving the following Python list

```
scores =
```

- $scores[2:5]$
- $scores[-3:-1]$
- $scores[:9:3]$
- $scores[4:]$
- $scores[-3:]$
- $scores[-1:]$

Chinese Proverb

I Hear & I Forget, I See & I Remember, I Do & I Understand

Adding Elements to a List

- To add elements to a list use the
  - *append*() method

```
my_list = []
my_list.append(48)
my_list.append(True)
print(f'my_list\n{my_list}')
# Output
# [48, True]
```

Remove an Element from a List

- To remove the last element from a list use the
  - *pop*() method

```
scores = [57, 58, 62, 55, 60, 63, 61, 68,
    60, 63]
x = scores.pop()
print(f'x = {x}')
print(f'scores after using pop()\n{scores}')
# Output
# x = 63
# scores after using pop()
# [57, 58, 62, 55, 60, 63, 61, 68, 60]
```

Remove an Element from a List

- To remove the element at specific index from a list use the
  - *del* method

```
scores = [57, 58, 62, 55, 60, 63, 61, 68, 60, 63]
del scores[4]
print(f'After removing element at index 4\n{scores}')
# Output
# After removing element at index 4
# [57, 58, 62, 55, 63, 61, 68, 60, 63]
```

List Concatenation

- To add two lists we use
  - + operator

```
list_1 = ['list','len',45,False]
list_2 = [1e3,0.5,(13,17)]
print(f'{list_1}+{list_2} is \n{list_1+
    list_2}')
# Output
# l['list', 'len', 45, False]+[1000.0, 0.5,
    (13, 17)] is
# ['list', 'len', 45, False, 1000.0, 0.5,
    (13, 17)]
```

List Membership

- To check if an element is in a list use
  - *in* operator

```
vancouver =  'Vancouver'                    -        False   'Surrey' 'Burnaby'
s  = 'Vancouver'
t = 'Burnaby'
print f'{s} is a member of the list?{s in vancouver}'
print f'{t} is a member of the list?{t in vancouver}'
# Output
# Vancouver is a member of the list?True
# Burnaby is a member of the list?False
```

List Sorting

- To sort a list in ascending order
  - *sort*() method

```
scores =
scores sort
print f'Sorting a list in ascending order\n{scores}'
# Output
# Sorting a list in ascending order
# [55, 57, 58, 60, 60, 61, 62, 63, 63, 68]
```

List Sorting

- To sort a list in descending order
  - $sort(reverse = True)$ method

```
scores =
scores sort reverse=True
print f'Sorting a list in descending order\n{scores}'
# Output
# Sorting a list in descending order
# [68, 63, 63, 62, 61, 60, 60, 58, 57, 55]
```

Python Tuple Data Structure

Tuples

- A tuple is simlar to a list stores
  - a collection of data as an ordered sequence
    - in a continuous memory locations
- A tuple can hold different data types
- To get the number of elements use the method *len*()

```
course_info =  'Computing for Data Analytics'  'CPSC 4800'
                True  False
print f'The tuple {course_info} has {len(course_info)} elements.'
```

Python Tuple Data Structure

Tuples

```
1 # empty tuple
2 tuple_0 = ()
3 # tuple with one element
4 tuple_1 = 'My single tuple',
5 # tuple with multiple elements
6 tuple_2 = 1,2,3,
7 # or
8 tuple_3 = (1,2,3)
```

Accessing Tuple Elements

- Elements of the tuple are accessed using positive indices
  - 0 index to access the first element `course_info[0]`
  - 1 index to access the second element `course_info[1]`
  - 2 index to access the third element `course_info[2]`
  - .....

```
print(f'The first element in the tuple is {course_info[0]}.')
print(f'The second element in the tuple is {course_info[1]}.')
print(f'The third element in the tuple is {course_info[2]}.')
```

Accessing Tuple Elements

- Elements of the tuple are accessed using negative indices
  - -1 index to access the last element `course_info[0]`
  - -2 index to access the second last element `course_info[1]`
  - -3 index to access the third last element `course_info[2]`
  - .....

```
print(f'The last element in the tuple is {course_info[-1]}.')
print(f'The second last element in the tuple is {course_info[-2]}.')
print(f'The third last element in the tuple is {course_info[-3]}.')
```

Python Tuple Data Structure

Tuple Content

- A tuple is a immutable object
  - its content CANNOT be changed

```
course_info    =
# Output
# TypeError: 'tuple' object does not support item assignment
```

Python Dictionary Data Structure

Dictionaries

- A dictionary is a key Python data structure that stores
  - a collection of key-value pairs (KVPs)
    - Generally, keys can be numbers or strings
- A dictionary can hold any Python data types for its values
- To get the number of elements use the method *len*()

y

```
course_info = {'title':'Computing for Data Analytics',
               'code':'CPSC 4800','credit':3, \
                'is_transferable':True,'fees':32.40}
print(f'The dictionary {course_info} has {len(course_info)} elements.')
```

Accessing Dictionary Elements

- Elements of the dictionary values are accessed using keys
  - To access the value of the title key `course_info['title']`

```
print(f'The value of the key title {course_info["title"]}.')
# Output
# The value of the key title Computing for Data Analytics.
```

Dictionary Content

- A dictionary is a mutable object
  - its content can be changed

```
course_info = {'title':'Computing for Data Analytics',
               'code':'CPSC 4800','credit':3, \
                'is_transferable':True,'fees':32.40}
course_info['fees'] = 50.95
print(f'The dictionary after making changes is\n{course_info}.')
# Output
# The dictionary after making changes is
# {'title': 'Computing for Data Analytics',
#  'code': 'CPSC 4800', 'credit': 3,
# 'is_transferable': True, 'fees': 50.95}.
```

Class Activity

- Giving the following Python dictionary

```
course_info = {'title':'Computing for Data Analytics',
               'code':'CPSC 4800','credit':3, \
                'is_transferable':True,'fees':32.40}
```

- What the value of code key?
- What is the value of credit key?
- What is the value of fees key?

Chinese Proverb

Tell Me & I Forget,
Teach Me & I Remember,
Involve Me & I Learn

Python Unpacking

Unpacking

- Python allows a list or tuple variable to be used as
  - a right-handside operand of an assignment

```
course_info = ('Computing for Data Analytics', 'CPSC 4800',3, \
                True,32.40)
title,code,credit,transferable,fees = course_info
print(f'(title,code,credit)=({title},{code},{credit})')
# Output
# (title,code,credit)=(Computing for Data Analytics,CPSC 4800,3)
```

Class Activity

- Giving the following Python list

```
vancouver = ['Vancouver',2632000, (49.25,-123.12),False,['Surrey','Burnaby']]
a,b,c = vancouver[:3]
```

- What are the values of a,b,c?

Chinese Proverb

I Hear & I Forget, I See & I Remember, I Do & I Understand

Class Activity

- Giving the following Python list

```
vancouver = ['Vancouver',2632000, (49.25,-123.12),False,['Surrey','Burnaby']]
a,b,c = vancouver[-3:]
```

- What are the values of a,b,c?

Chinese Proverb

I Hear & I Forget, I See & I Remember, I Do & I Understand