1 Python Data Types and Structures

For COMSC 132

Sam Bowne

Aug 12, 2024

Python Setup

- Start at **samsclass.info**
- Click **COMSC 132**
- Click **Projects**

Prepare a Python Environment

Do at least one of these

VP 11: Python 3 on Google Colab (10)
VP 10: Python 3 Installed Locally (10 extra)
ML 125: Jupyter Notebook on a Mac M1 (10 extra)

Dynamic Typing

- Python figures out data types at runtime

```
p = "Hello India"
q = 10
r = 10.2
print(type(p))
print(type(q))
print(type(r))
print(type(12+31j))
```

```
<class 'str'>
<class 'int'>
<class 'float'>
<class 'complex'>
```

Basic Data Types

Basic Data Types

- Numeric
- Boolean
- Sequences
  - String
  - Range
  - List
  - Tuples

Numeric Data Types

- **int**
  - Integers like 45 or -25
- **Float**
  - With a decimal point, like 3.14
- **Complex**
  - a + *i*b
  - a and b are floats

Boolean Data

- Either **true** or **false**
- Any non-zero value is **true**
- Zero is **false**

```
print(bool(22))
print(type(bool(22)))
print(type(True))
print(type(False))
```

```
True
<class 'bool'>
<class 'bool'>
<class 'bool'>
```

Sequences

- **Strings**
  - Immutable sequence of characters
  - In single, double, or triple quotes

```
str1 = 'foo'
str2 = 'bar'
print(str1 + str2)

# creates a new string object
str1 = str1 + str2
print(str1)

print(str1[0])
str1[0] = "A"
```

```
foobar
foobar
f
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-f7c0ee880a99> in <cell line: 10>()
      8
      9 print(str1[0])
---> 10 str1[0] = "A"

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

Sequences

- **Range**
  - Immutable sequence of numbers

```python
print(list(range(10)))
print(range(10))

print(range(1,10,2))
print(list(range(1,10,2)))

print(list(range(20,10,-2)))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
range(0, 10)
range(1, 10, 2)
[1, 3, 5, 7, 9]
[20, 18, 16, 14, 12]
```

Lists

- Can store duplicate values
- Elements can be of different types
- Mutable

```
[11] a = ['food', 'bus', 'apple', 'queen']
     print(a)

     mylist  = [10, "India", "world", 8]
     print(mylist)

     # accessing elements in list.
     print(mylist[1])

     mylist[0] = 'frog'
     print(mylist)
```

```
['food', 'bus', 'apple', 'queen']
[10, 'India', 'world', 8]
India
['frog', 'India', 'world', 8]
```

Membership, Identity, and Logical Operations

- Membership operators
- Identity operators
- Logical operators

Membership operators

- **in** and **not in**
- Test to see if a value is in an object
  - Such as a string, list, or tuple

```
[13] mylist  = [10, "India", "world", 8]
     if 10 in mylist:
       print("10 found in mylist")
     else:
       print("10 not found in mylist")
```

```
10 found in mylist
```

Identity operators

- **is** and **is not**
- Checks to see if two variables refer to the same object
- Not the same as equality **==**
  - Checks whether two variables have the same value

```
[15] Firstlist = []
     Secondlist = []
     if Firstlist == Secondlist:
         print("Both are equal")
     else:
         print("Both are not equal")
     if Firstlist is Secondlist:
         print("Both variables are pointing to the same object")
     else:
         print("Both variables are not pointing to the same object")
     thirdList = Firstlist
     if thirdList is Secondlist:
         print("Both are pointing to the same object")
     else:
         print("Both are not pointing to the same object")
```

```
Both are equal
Both variables are not pointing to the same object
Both are not pointing to the same object
```

Logical operators

- **AND**, **OR**, and **NOT**
- Combine logical operators

```
a = 32
b = 132
if a > 0 and b > 0:
  print("Both a and b are greater than zero")
else:
  print("At least one variable is less than 0")
```

```
Both a and b are greater than zero
```

Tuples

- Immutable
- Data is ordered
- Duplicate values are allowed
- Elements can be of different types

```
[17] my_tuple = ("Shyam", 23, True, "male")
     print(my_tuple[0])
     my_tuple[0] = 1
```

```
Shyam
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-17-b29686f7be67> in <cell line: 3>()
      1 my_tuple = ("Shyam", 23, True, "male")
      2 print(my_tuple[0])
----> 3 my_tuple[0] = 1

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

Complex Data Types

Complex Data Types

- Dictionary
- Set
- Frozenset

Dictionary

- A collection of **key:value** pairs
- Unordered
- Key is immutable
- Value can be any object

```
my_dict = {'1': 'data',
           '2': 'structure',
           '3': 'python',
           '4': 'programming',
           '5': 'language'
}
print(my_dict)
print(my_dict['2'])
```

```
{'1': 'data', '2': 'structure', '3': 'python', '4': 'programming', '5': 'language'}
structure
```

Sets

- Unordered collection of objects
- Iterable, mutable, and has unique objects
- Order is not defined

```
x1 = set(['and', 'python', 'data', 'structure'])
print(x1)
print(type(x1))
x2 = {'and', 'python', 'data', 'structure'}
print(x2)
print(type(x2))
```

```
{'and', 'python', 'data', 'structure'}
<class 'set'>
{'and', 'python', 'data', 'structure'}
<class 'set'>
```

Immutable Sets

- Cannot be changed after creation

```
x = frozenset(['data', 'structure', 'and', 'python'])
print(x)
```

```
frozenset({'and', 'python', 'data', 'structure'})
```

Collections module

- Provides **containers**
  - named tuple
  - deque
  - defaultdict
  - ChainMap
  - Counter
  - UserDict, UserList, UserString

Kahoot!

Ch 1