ED5340 - Data Science: Theory and Practise

L11 - Classes Container and Inheritance, Iterators and generators

Ramanathan Muthuganapathy (https://ed.iitm.ac.in/~raman)
Course web page: https://ed.iitm.ac.in/~raman/datascience.html
Moodle page: Available at https://courses.iitm.ac.in/

Class containership

‘has a’ relationship - L11_Container_Ex1.py

- Faculty has a department
- Student has a department

Classes - Inheritance

‘like a’ relationship - classDerived.py

- Base class
- Derived class - derived from base class
- Base class - called as super class or parent class
- Derived class - subclass or child class
- Syntax: class Derived(Base):

Inheritance - key points

classAccessibility.py

- Construction of an object - from base to derived
- Same thing for constructor - use super( ).__init__( )
- Derived class can access data / methods of the base class
- var, _var, __var (similar to public, protected and private in C++, in practise, only public and protected)
- same method in both derived and base - derived class method.
- super( ).baseclassmethod( ) or baseclassname.baseclassmethod(self)

Types of Inheritance

L11_MultiLevelInheritance.py and L11_MultipleInheritance.py

- Multi-level inheritance - Derived1(Base), Derived2(Derived1)
- Multiple inheritance - Derived from two base classes.
    - class Derived(Base1, Base2)

HW: Abstract classes - What are they?

Iterators

```
lst = [10, 20, 30]

for ele in lst:

    print(ele)
```

- for loops call __iter__( ) returns an iterator object
- The iterator object has a method __next__( )
- In __next__( ), StopIteration exception is raised

Generators

L11_generators.py

- Generators are functions that create iterators
- Uses yield instead of return
- It remembers the last state (something similar to a static one)
- When next( ) is called, it resumes where it had left off
- Generators can replace the class-based iterators.
- __iter__( ), __next__( ) and StopIteration code is created automatically

Generator expression

L11_generators.py

- Similar to comprehension!
- Creates a generator on the fly without using yield statement
- ( ) (instead of [ ] or { })
- Takes less memory than a list comprehension