Dinasarsam

Exercise: Learning from Object Oriented Programming Code

Biopython SeqRecord Class Analysis: Answer by hand except where it says **copy/paste**

Part I: Imported modules

1) Describe the purpose of the typing module:

The typing module provide type hints. enabling static type checking. Also, it can specify the expected types of variables, function parameters and return values.

2) **Copy/paste** a code snippet example of the use of the typing module in the code. (Check out the __init__):

stands for initialize

name of base class.

This code from Seq.py file on Canvas

```
def __init__(self, data):
    """Create a MutableSeq object."""
    if isinstance(data, bytearray):
        self._data = data
    elif isinstance(data, bytes):
        self._data = bytearray(data)
    elif isinstance(data, str):
        self._data = bytearray(data, "ASCII")
    elif isinstance(data, MutableSeq):
        self._data = data._data[:]  # Take a copy
    elif isinstance(data, Seq):
        # Make no assumptions about the Seq subclass internal storage
        self._data = bytearray(bytes(data))
    else:
        raise TypeError(
            "data should be a string, bytearray object, Seq object, or a "
            "MutableSeq object"
```

Explain what this code does:

It allows the class to accept a wide range of sequence input types. The strings or other sequence objects and store the sequence internally on a bytearray to modify.

3) Describe the purpose of the doctest module:

It's to insure that the code is correct. Also, to improve both the documentation and the quality of the code by testing real examples within the docstring.

4) **Copy/paste** an example of a doctest in the code:

↳ see next page

Explain what this code does:

The code defines a factorial function, then tests it using the Doctest framework. The results will be Pass or fail.

```
#define DOCTEST_CONFIG_IMPLEMENT_WITH_MAIN
#include "doctest.h"

int factorial(int number) { return number <= 1 ? number : factorial(number - 1) * number; }

TEST_CASE("testing the factorial function") {
    CHECK(factorial(1) == 1);
    CHECK(factorial(2) == 2);
    CHECK(factorial(3) == 6);
    CHECK(factorial(10) == 3628800);
}
```

→ doctest code from github

Where are the doctests executed?

They executed directly with Python modules, functions, classes, or methods. A test is executed

Part II: Identify parts of the code using OOP Terms:

1) What kind of Class is SeqRecord? It is data container class used to store biological sequences
2) On what line does the SeqRecod start and what is it called?
   LINE NUMBER: NAME OF METHOD: → SeqRecord
   ← The source code
3) What data types are the following attributes (write next to the names):
   a. Id -> store identifier of the sequence as a string
   b. Name -> Store the name of sequence as a string.
   c. Features -> store a list of Bio.SeqFeature object
      ↳ It shows biological feature like gene. Exon.
4) Find an example of an instance of the class in one of the docstrings. (__getitem__ has one) and **copy/paste** it here:

   The example I found is: → from Seq.py

```
Class MyList:
 def __getitem__(self, index):
    """Return a subsequence as a single letter or as a sequence object.

    If the index is an integer, a single letter is returned as a Python
    string:

    >>> seq = Seq('ACTCGACGTCG')
    >>> seq[5]
    'A'
    >>> seq[5:8]
    Seq('ACG')
    >>> mutable_seq = MutableSeq('ACTCGACGTCG')
    >>> mutable_seq[5:8]
    MutableSeq('ACG')
    """
    if isinstance(index, numbers.Integral):
      # Return a single letter as a string
```

```
            return chr(self._data[index])
        else:
            # Return the (sub)sequence as another Seq/MutableSeq object
            return self.__class__(self._data[index])
```

5) What attributes are assigned values when this instance is created? Write them here:

Seq: → sequence data
id: → the ID seq.
name: → name of seq.
description → The description of the seq.
features: empty list [ ]
annotations: empty dictionary { }

6) What ones are not assigned? Write them here:

- If there isn't a sequence features such as gene, exon are provided this attribute remains an empty list.
- annotations → { } → empty list

7) What is @property and @overload?

- Property → is used to define getter methods for attributes in a class. Also, allow accessing the object
- overload → It helps with accept different argument types and return different types, without providing the actual implementation.

Part III: Investigate some methods

1) How would you describe the __str__ method in OOP terminology?

__str__ → special method that defines how an object should be represented as a string.

2) What would the following output be based on the __str__ function?

```
>>> record = SeqRecord(Seq("SNACKATTACK"), id="CHPS_1001", name=" BBQ5",
description="fried junk food")

>>> print(record)
```

Write answer here:

When we call print(record) python calls the __str__ method of that object. The __str__ method for seqRecord is defined in Biopython

3) What built-in functions or operators do the following effect? Write the answer next to the method:
   a. __len__ → Return the length of the object.
   b. __lt__ → less than operator. (<)
   c. __eq__ → equal ==
   d. __ne__ → not equal → !=
   e. __add__ → To add two amount +

4) Name three methods that do not affect built-in methods or operators and explain what they do:

1- display() or show() → display or print object info.
2- reset() → To reset the attributes of an object.
3- show() → To print object info in a custom format.

5) What is the purpose of if __name__ == '__main__":

To execute the code depending on how the script is being used.

6) Will this code run if you import SeqRecord? Explain your answer:

It will not run because python has special variable __name__ of the module not "main"
- For the code to run, it needs file to executed directly.