Programming for Biologists

Python Objects and Biopython

`https://biopython.org/`

Learning Objectives

- Python Classes
- Inheritance of Classes
- Object oriented packages
- Example use of Biopython objects
- Code Review

Python's Object Paradigm

- Python's built-in data types have their own functions that can be called on it.

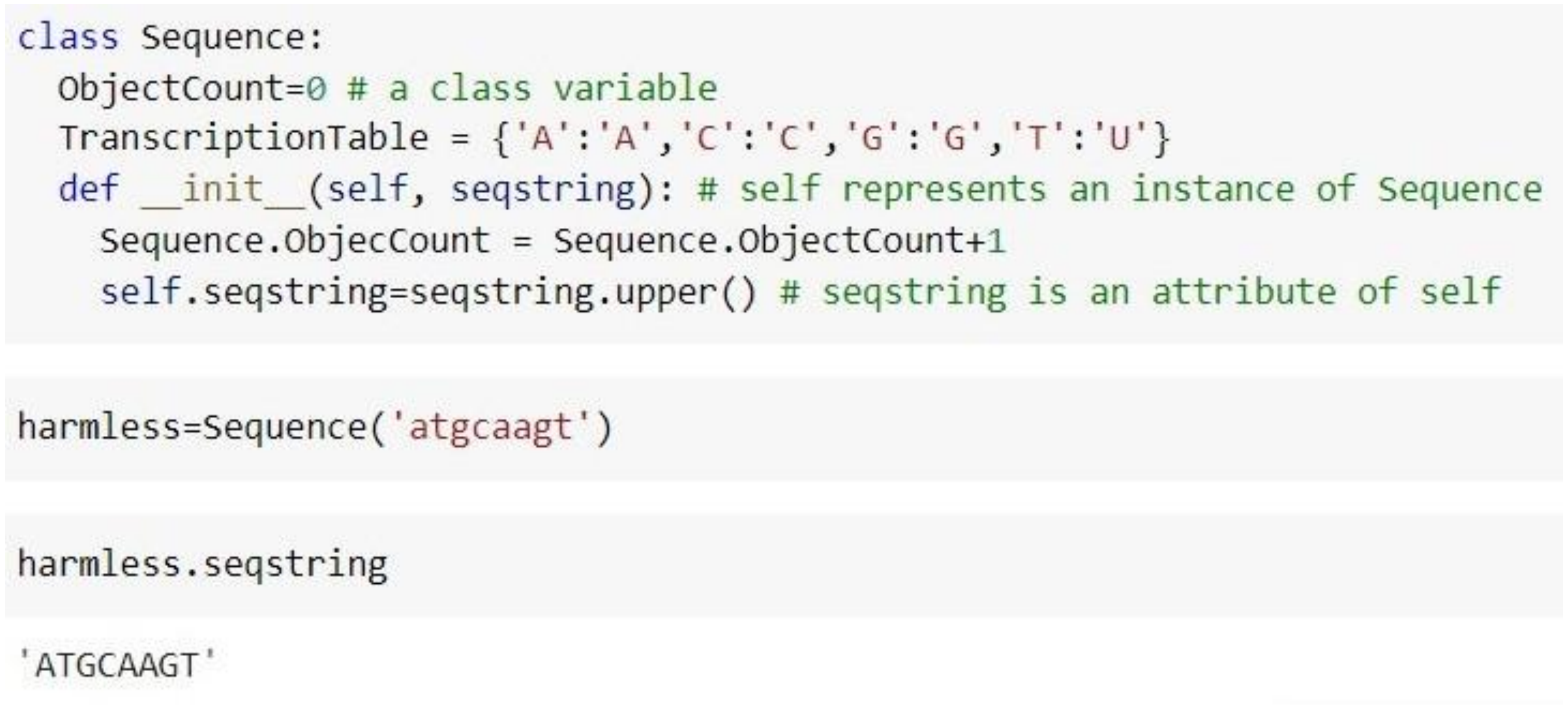
- A Class is a user-defined data type with its own specific attributes and functions:
  - An instance or object template.
  - Any object that can be abstracted can be a class:

    e.g. person, mammal, animal; sequence, plasmid, protein

Python's Object Paradigm - Terminology

- A particular instantiation of a class is called an instance ("object" is essentially used as a synonym for "instance").
- Class instances have their own characteristics (attributes or properties); these are called data attributes.
- Methods (also called member functions) are functions that belong to a specific data type or class and define how objects derived from that class "behave".

Creating Python Classes



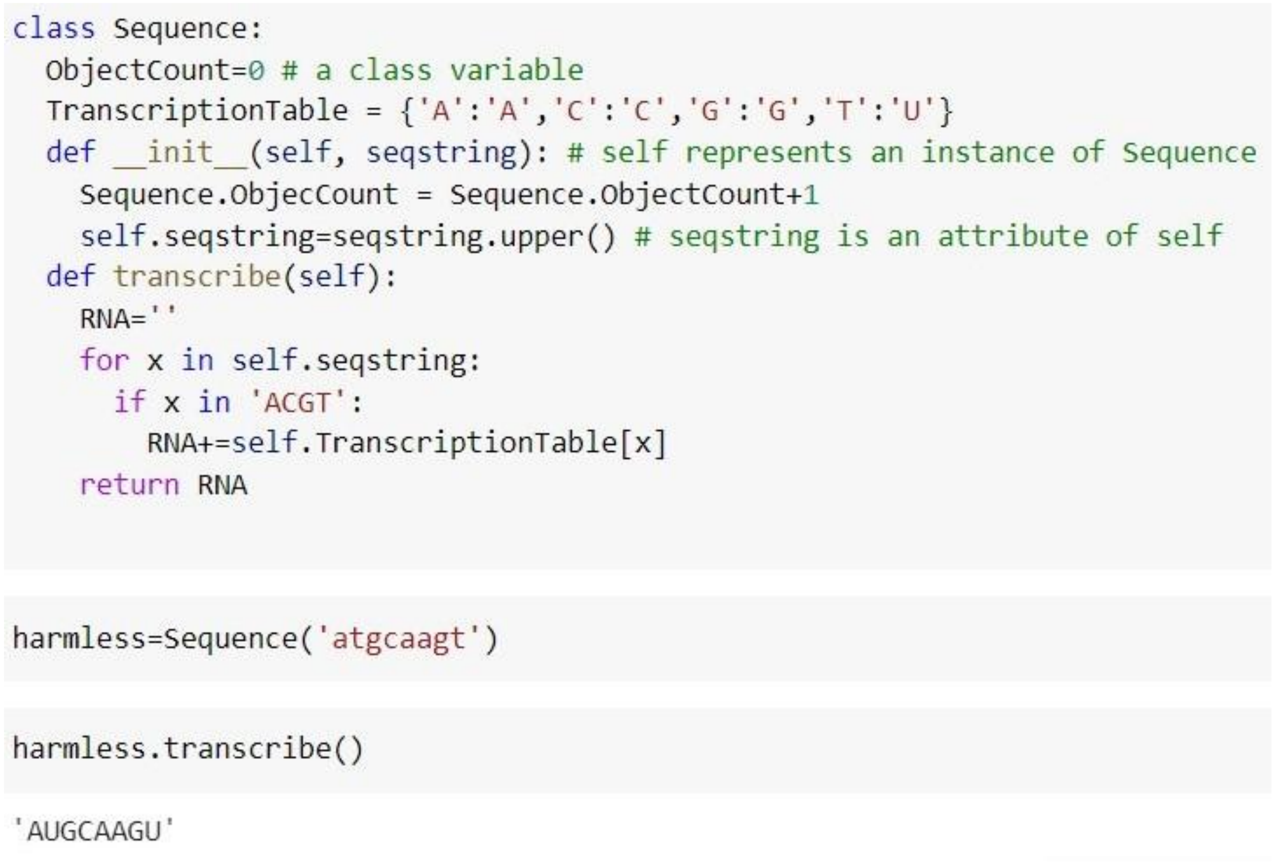
```
class Sequence:
  ObjectCount=0 # a class variable
  TranscriptionTable = {'A':'A','C':'C','G':'G','T':'U'}
  def __init__(self, seqstring): # self represents an instance of Sequence
    Sequence.ObjecCount = Sequence.ObjectCount+1
    self.seqstring=seqstring.upper() # seqstring is an attribute of self
```

```
harmless=Sequence('atgcaagt')
```

```
harmless.seqstring
```

```
'ATGCAAGT'
```

Creating Python Classes



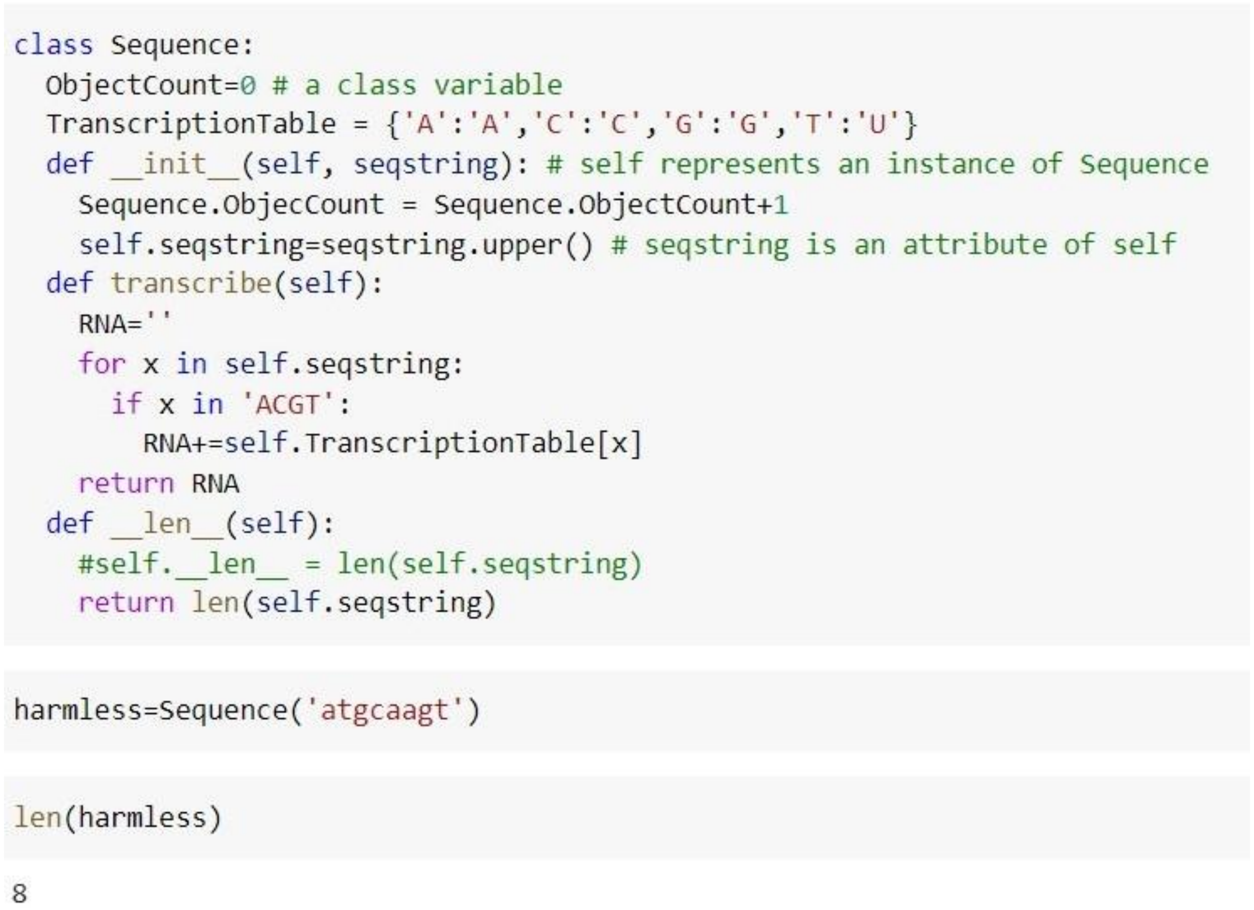
```python
class Sequence:
  ObjectCount=0 # a class variable
  TranscriptionTable = {'A':'A','C':'C','G':'G','T':'U'}
  def __init__(self, seqstring): # self represents an instance of Sequence
    Sequence.ObjecCount = Sequence.ObjectCount+1
    self.seqstring=seqstring.upper() # seqstring is an attribute of self
  def transcribe(self):
    RNA=''
    for x in self.seqstring:
      if x in 'ACGT':
        RNA+=self.TranscriptionTable[x]
    return RNA
```

```python
harmless=Sequence('atgcaagt')
```

```python
harmless.transcribe()
```

```
'AUGCAAGU'
```

Operator overloading



```
class Sequence:
  ObjectCount=0 # a class variable
  TranscriptionTable = {'A':'A','C':'C','G':'G','T':'U'}
  def __init__(self, seqstring): # self represents an instance of Sequence
    Sequence.ObjecCount = Sequence.ObjectCount+1
    self.seqstring=seqstring.upper() # seqstring is an attribute of self
  def transcribe(self):
    RNA=''
    for x in self.seqstring:
      if x in 'ACGT':
        RNA+=self.TranscriptionTable[x]
    return RNA
  def __len__(self):
    #self.__len__ = len(self.seqstring)
    return len(self.seqstring)
```

```
harmless=Sequence('atgcaagt')
```

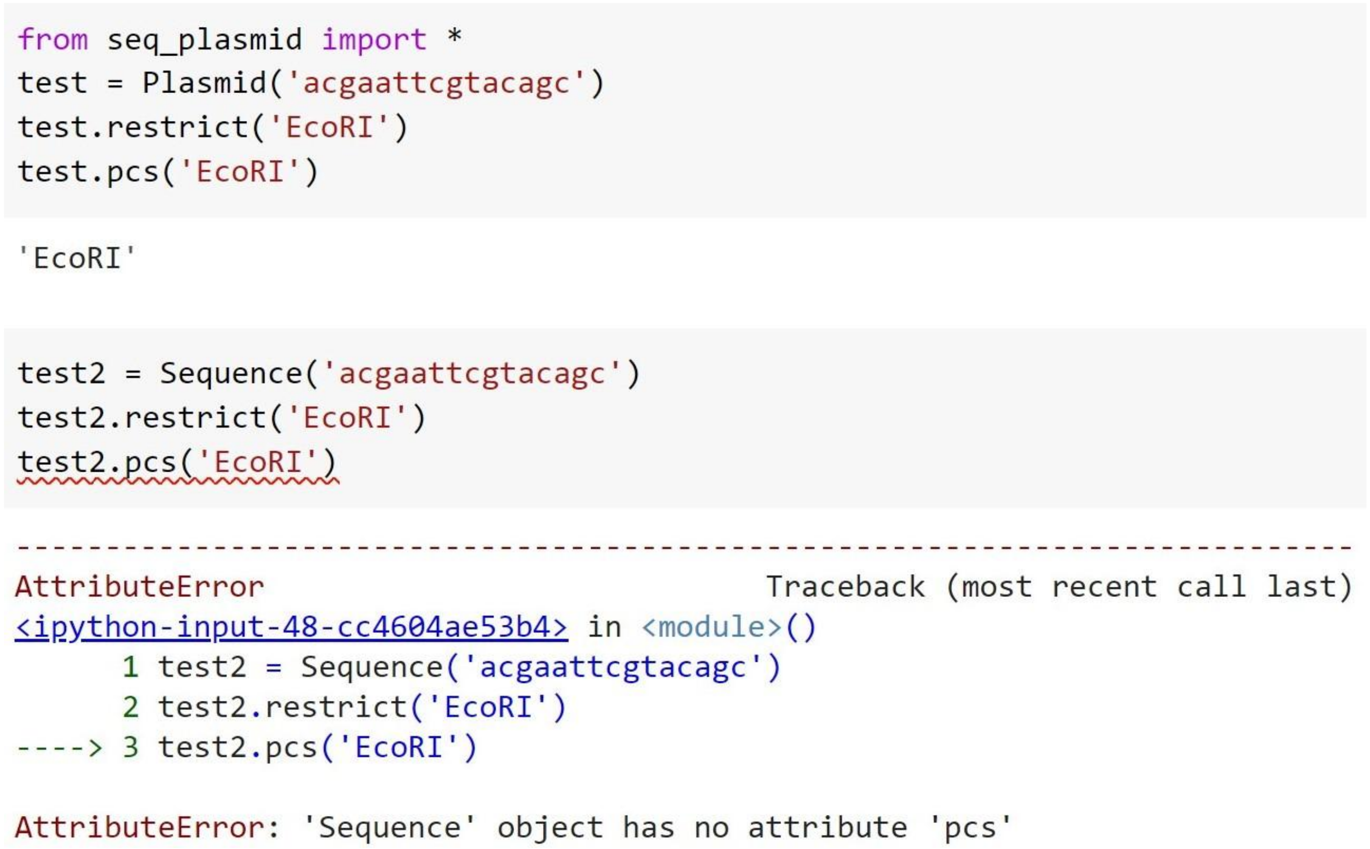
```
len(harmless)
```

```
8
```

Inheritance

- Derived classes can inherit properties and methods from a parent class, but implement attributes and methods specific to their subclass.
- For example, the 'seq_plasmid.py' module defines two classes, Sequence and Plasmid:
    - Sequence contains two methods, 'transcribe', and 'restrict'.
    - Plasmid inherits these methods, but also contains the methods 'pcs' and 're_in_pcs'.

Inheritance example



```
from seq_plasmid import *
test = Plasmid('acgaattcgtacagc')
test.restrict('EcoRI')
test.pcs('EcoRI')
```

```
'EcoRI'
```

```
test2 = Sequence('acgaattcgtacagc')
test2.restrict('EcoRI')
test2.pcs('EcoRI')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-48-cc4604ae53b4> in <module>()
      1 test2 = Sequence('acgaattcgtacagc')
      2 test2.restrict('EcoRI')
----> 3 test2.pcs('EcoRI')

AttributeError: 'Sequence' object has no attribute 'pcs'
```

Biopython - https://biopython.org/

- The BioPython package contains a large number of modules that implement specialized classes for biological sequence analysis.
- Using them typically entails creating one or more instances of a particular object type, and then calling different methods on these.
- Today we will learn about BioPython modules for manipulating sequence data.
- Documentation for BioPython modules can be found online.
- To install type: **!pip install biopython**

Biopython Example

```
from Bio.Seq import Seq
my_dna = Seq("AGTACACTGGT")
```

```
my_dna.reverse_complement()
```

```
Seq('ACCAGTGTACT')
```

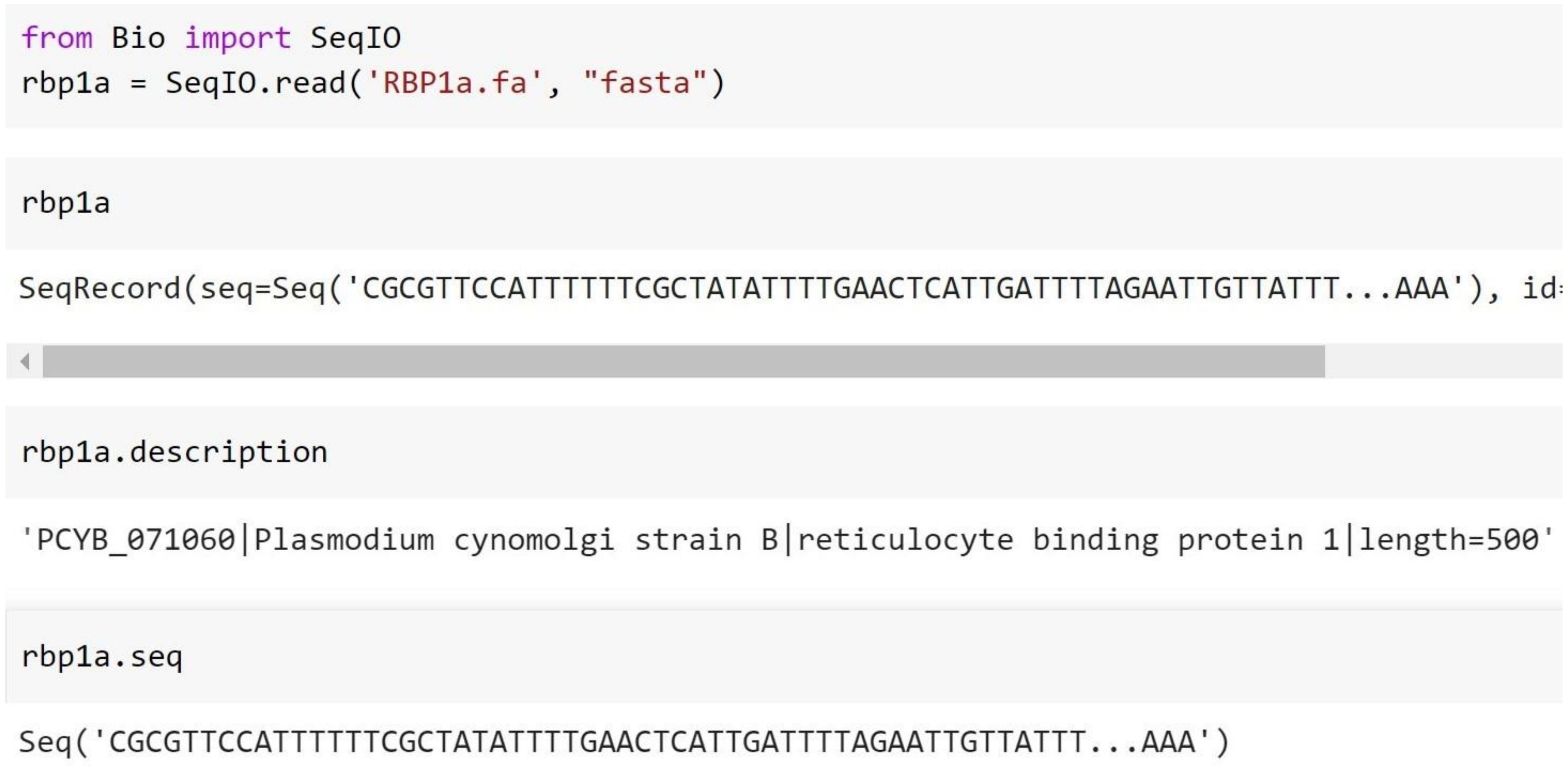
```
my_dna.transcribe()
```

```
Seq('AGUACACUGGU')
```

```
my_dna[:(int(len(my_dna) / 3)*3)].translate()
```

```
Seq('STL')
```

Biopython - SeqIO - Read file



```
from Bio import SeqIO
rbp1a = SeqIO.read('RBP1a.fa', "fasta")
```

```
rbp1a
```

```
SeqRecord(seq=Seq('CGCGTTCCATTTTTTCGCTATATTTTGAACTCATTGATTTTAGAATTGTTATTT...AAA'), id
```

```
rbp1a.description
```

```
'PCYB_071060|Plasmodium cynomolgi strain B|reticulocyte binding protein 1|length=500'
```

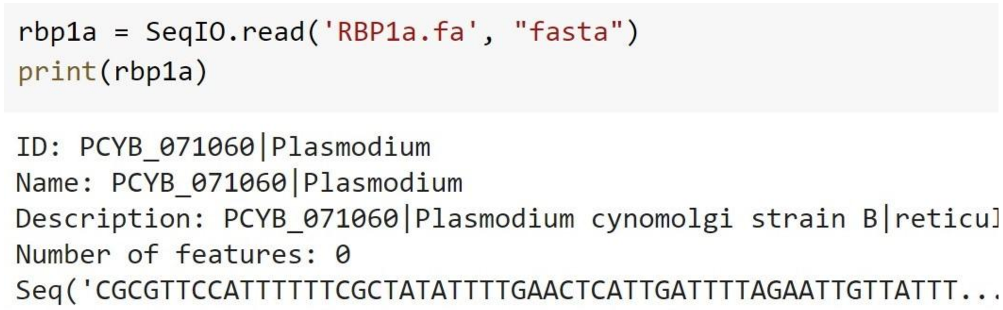
```
rbp1a.seq
```

```
Seq('CGCGTTCCATTTTTTCGCTATATTTTGAACTCATTGATTTTAGAATTGTTATTT...AAA')
```

BioPython SeqRecord objects

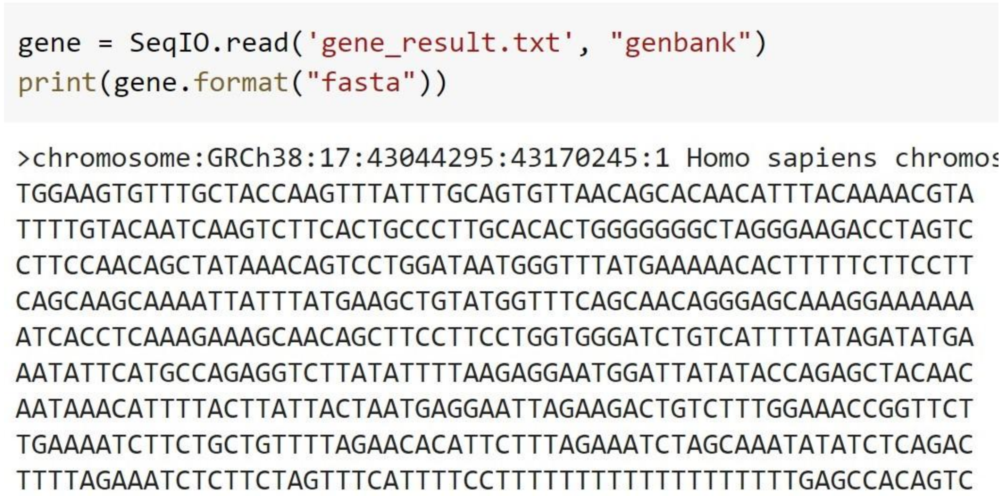
- If we want to know more about a sequence, like its name, ID, description, etc., we need a SeqRecord object, which stores a Seq object with its associated metadata
- More information can be found here https://biopython.org/wiki/SeqRecord

Biopython SeqRecords



```python
rbp1a = SeqIO.read('RBP1a.fa', "fasta")
print(rbp1a)
```

```
ID: PCYB_071060|Plasmodium
Name: PCYB_071060|Plasmodium
Description: PCYB_071060|Plasmodium cynomolgi strain B|reticul
Number of features: 0
Seq('CGCGTTCCATTTTTTCGCTATATTTTGAACTCATTGATTTTAGAATTGTTATTT...
```

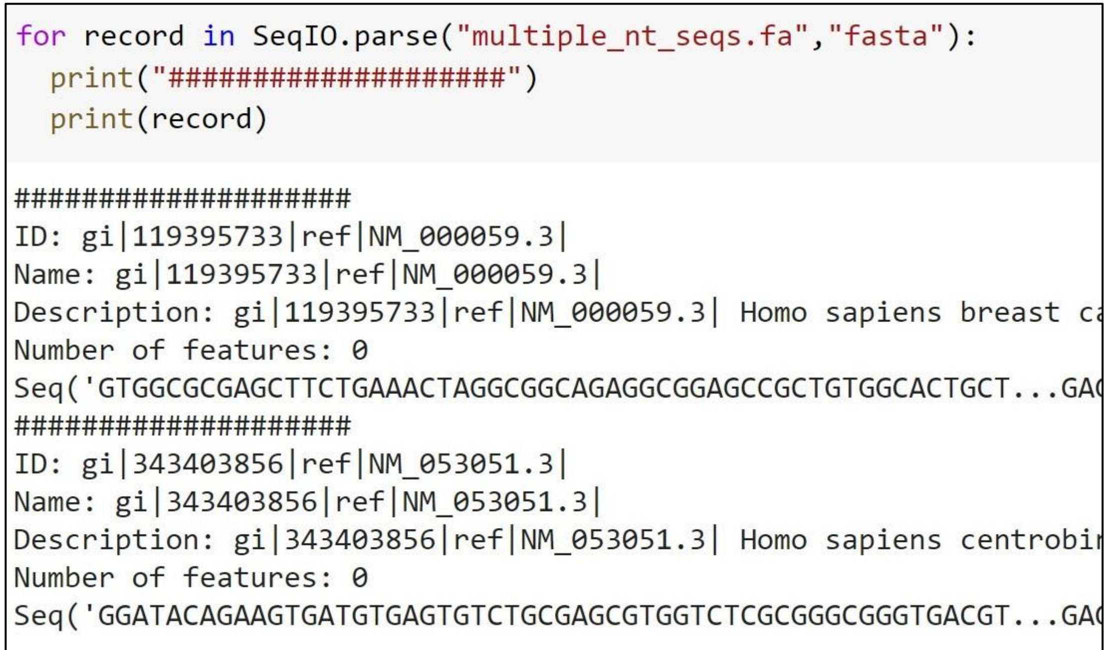


```python
gene = SeqIO.read('gene_result.txt', "genbank")
print(gene.format("fasta"))
```

```
>chromosome:GRCh38:17:43044295:43170245:1 Homo sapiens chromos
TGGAAGTGTTTGCTACCAAGTTTATTTGCAGTGTTAACAGCACAACATTTACAAAACGTA
TTTTGTACAATCAAGTCTTCACTGCCCTTGCACACTGGGGGGGCTAGGGAAGACCTAGTC
CTTCCAACAGCTATAAACAGTCCTGGATAATGGGTTTATGAAAAACACTTTTTCTTCCTT
CAGCAAGCAAAATTATTTATGAAGCTGTATGGTTTCAGCAACAGGGAGCAAAGGAAAAAA
ATCACCTCAAAGAAAGCAACAGCTTCCTTCCTGGTGGGATCTGTCATTTTATAGATATGA
AATATTCATGCCAGAGGTCTTATATTTTAAGAGGAATGGATTATATACCAGAGCTACAAC
AATAAACATTTTACTTATTACTAATGAGGAATTAGAAGACTGTCTTTGGAAACCGGTTCT
TGAAAATCTTCTGCTGTTTTAGAACACATTCTTTAGAAATCTAGCAAATATATCTCAGAC
TTTTAGAAATCTCTTCTAGTTTCATTTTCCTTTTTTTTTTTTTTTTTTTGAGCCACAGTC
```

Parsing sequence records

- SeqIO.read - works only on a single sequence record
- SeqIO.parse - iterates through a series of sequence records

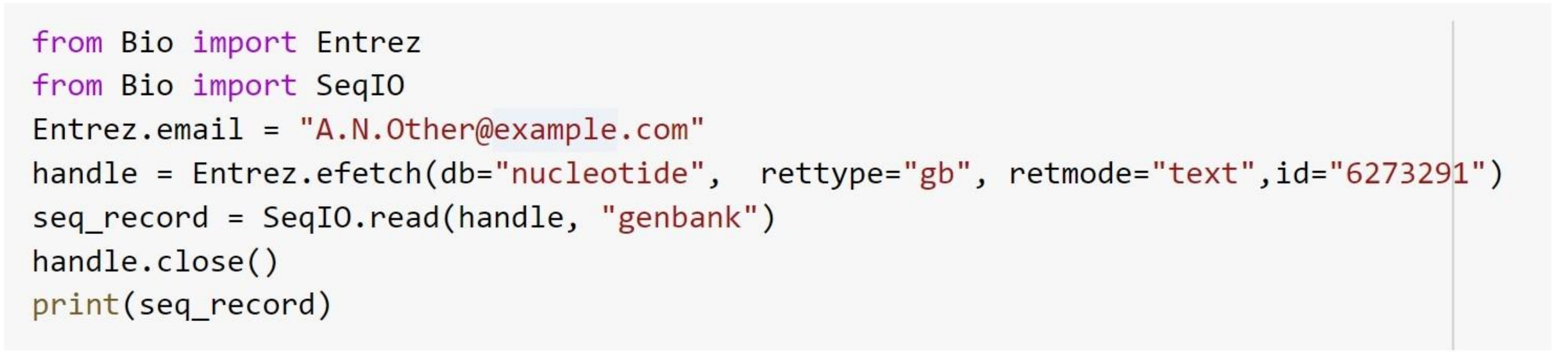


```
for record in SeqIO.parse("multiple_nt_seqs.fa","fasta"):
  print("####################")
  print(record)
```

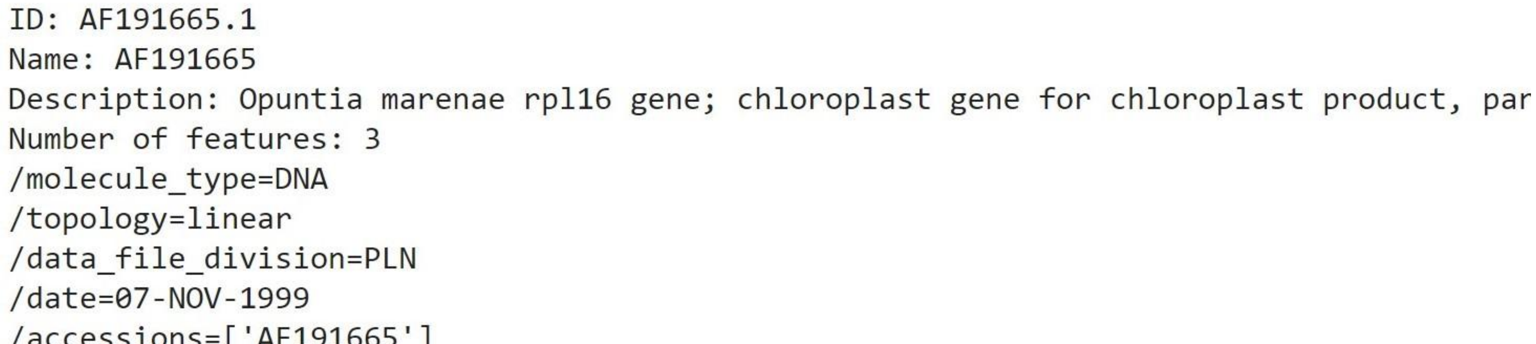
```
####################
ID: gi|119395733|ref|NM_000059.3|
Name: gi|119395733|ref|NM_000059.3|
Description: gi|119395733|ref|NM_000059.3| Homo sapiens breast ca
Number of features: 0
Seq('GTGGCGCGAGCTTCTGAAACTAGGCGGCAGAGGCGGAGCCGCTGTGGCACTGCT...GA
####################
ID: gi|343403856|ref|NM_053051.3|
Name: gi|343403856|ref|NM_053051.3|
Description: gi|343403856|ref|NM_053051.3| Homo sapiens centrobi
Number of features: 0
Seq('GGATACAGAAGTGATGTGAGTGTCTGCGAGCGTGGTCTCGCGGGCGGGTGACGT...GA
```

Retrieving SeqRecord objects from Entrez

- Download the sequence directly from site.



```python
from Bio import Entrez
from Bio import SeqIO
Entrez.email = "A.N.Other@example.com"
handle = Entrez.efetch(db="nucleotide",  rettype="gb", retmode="text",id="6273291")
seq_record = SeqIO.read(handle, "genbank")
handle.close()
print(seq_record)
```



```
ID: AF191665.1
Name: AF191665
Description: Opuntia marenae rpl16 gene; chloroplast gene for chloroplast product, par
Number of features: 3
/molecule_type=DNA
/topology=linear
/data_file_division=PLN
/date=07-NOV-1999
/accessions=['AF191665']
```