

```
>>> from Bio.Seq import Seq
>>> from Bio.Alphabet import IUPAC
>>> seq1 = Seq('CCGTAGCATTAGCCGATTGTGATTANAT', IUPAC.ambiguous_dna)
>>> len(seq1) # value is 21
```

```
class Seq:
```

```
    def __init__(self, data, alphabet = Alphabet.generic_alphabet):
        # Enforce string storage
        assert type(data) == type("") # must use a string
```

```
        self.data = data
        self.alphabet = alphabet
```

```
    #The following methods are operator/function overloading
```

```
    def __repr__(self): #Overloading print function
        return "%s(%s, %s)" % (self.__class__.__name__,
                                repr(self.data),
                                repr(self.alphabet))
```

```
    def __len__(self): return len(self.data) #length function
```

```
    def __getslice__(self, i, j): #overload slice function
        i = max(i, 0); j = max(j, 0)
        return Seq(self.data[i:j], self.alphabet)
```

```
    def tostring(self): #Class method
        return self.data
```

Sequence Utilities:

```
>>> from Bio import utils #A few examples
utils.transcribe(seq) #This bunch of functions require Seq objects (see previous slides)
utils.ungap(seq)
utils.back_transcribe(seq)
```

```
>>> utils.transcribe(seq1)
Seq('CCGUAGCAUUAGCCGAUUGUGAUUANAU', IUPACAmbiguousRNA())
```

```
>>> from Bio import sequtils #Some more examples
sequtils.complement(seq) #Sequtils do not require Seq objects but can work on them
sequtils.antiparallel(seq)
sequtils.six_frame_translations(seq, genetic_code = 1)
sequtils.GC(seq)
```

```
>>> sequtils.antiparallel(seq1)
'ATNTAATCACAATCGGCTAATGCTACGG'
```

Parsing Data Files

What is parsing?

Why is it necessary?

How is it useful?

Different File Formats

FASTA

```
>gi|16753181|gb|AY040286.1| Dendroctonus adjunctus isolate Hlt12 cytochrome oxidase subunit 1 (CO1) gene,  
TTTTAAGAAGAATTATTGATAAAGGAGCAGGAAGCTGGTTGAACAGTCTATCCCCCGTTAGCCTCTAATAT  
TTCTCATGAAGGCTCCTCAGTAGATTGTGCAATTTTTAGGCTTCATATAGCTGGTATTTCTCTATTTTA  
GGGGCTATTAATTTTATTTCTACAATTATAAATATAGCCCCTTTAGGGATAAAATTAGATCGATTAACAT
```

GENBANK

```
LOCUS     AY040286             1129 bp  DNA   linear  INV 08-MAR-2002  
DEFINITION Dendroctonus adjunctus isolate Hlt12 cytochrome oxidase subunit 1  
            (CO1) gene, partial cds; mitochondrial gene for mitochondrial  
            product.  
ACCESSION  AY040286  
VERSION    AY040286.1 GI:16753181  
KEYWORDS   .
```

FASTA RECORD PARSER

```
def fasta_parser(fasta_file):
    "One argument (fasta_file) is a string file name for a fasta text file"
    parser = Fasta.RecordParser() #Create a specific BioPython fasta parser
    file = open(fasta_file, 'r')
    iterator = Fasta.Iterator(file,parser) #iterator goes through file


    """This while loop iterates through each line of the file until no
    more entries are found. Returns the data to self.fasta_data"""
    while 1:
        cur_record = iterator.next() #Gets each record one at a time

        if cur_record is None:
            break #Ends while loop at break

        print cur_record.title
        print cur_record.sequence
```

FASTA file from NCBI (on D drive)

accession numbers



```
>gi|16507255|gb|AF067986.2| Dendroctonus frontalis isolate Hlt14 cytochrome oxidase subunit 1 (COI)
TTTAGCTTCTAATATTTCCCTAGAAGGTTCTTCAGTAGATTGTGCAATTTTTAGACTTCATATAGCAGGA
ATCTCATCTATTTTAGGAGCTATCAATTTTATCTCAACAATTATAAATATAGCCCCTTTAGGTATAAAAT
>gi|16753181|gb|AY040286.1| Dendroctonus adjunctus isolate Hlt12 cytochrome oxidase subunit 1 (CO1)
TTTTAAGAAGAATTATTGATAAAGGAGCAGGAAGCTGGTTGAACAGTCTATCCCCGTTAGCCTCTAATAT
TTCTCATGAAGGCTCCTCAGTAGATTGTGCAATTTTTAGGCTTCATATAGCTGGTATTTCTCTATTTTA
```

Calling the function

```
>>> fasta_parser('D://test_fasta.txt')
>gi|16507255|gb|AF067986.2| Dendroctonus frontalis...
TTTAGCTTCTAATATTTCCCTAGAAGGTTCTTCAGTAG...
```