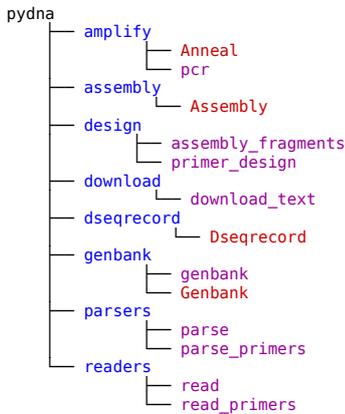




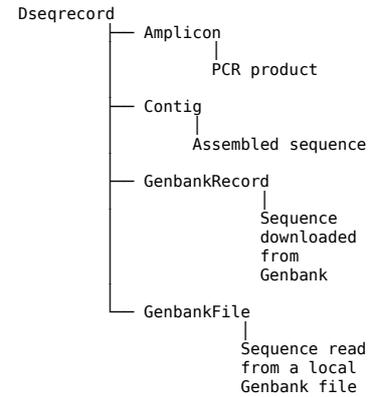
# Pydna cheat sheet

## 1. Important modules, functions and Classes

```
from pydna.module import function
from pydna.module import Class
```



Useful subclasses of Dseqrecord:



## 2. Set email address and other global settings in pydna.

```
import pydna
pydna.open_config_folder()
```

edit and save the file called “pydna.ini” in the folder that opens.

## 3. Establish sequence(s) in pydna.

sequence is in Genbank with known accession number: `seq = genbank(“L09137”)`

read from local file: `seq = read(“myseq.gb”)`

read a list of multiple sequences from local file: `seq = parse(“mysequences.gb”)`

create a **Dseqrecord** object directly: `seq = Genbank(“gatc”)`

## 4. Cut with restriction enzymes.

```
From Bio.Restriction import BamHI
list_of_seqs = seq.cut(BamHI)
seq_bam = plasmid.linearize(BamHI)
```

The linearize method works only on circular sequences and allow only one resulting fragment. The cut method returns a list. To cut with more enzymes, add them separated by comma.

## 5. Make a circular sequence from a linear.

```
circular_seq = linear_seq.looped()
```

Note that `linear_seq` has to have compatible sticky or blunt ends.

## 6. Saving the sequence to a file:

```
seq.write(“filename.ext”) (Default format is the Genbank flat file format)
```



## Pydna cheat sheet

### 7. Design primers for a sequence

```
amplicon = primer_design(seq)
```

The amplicon object describes a PCR reaction. Access the primers like this:

```
forwardprimer = amplicon.forward_primer OR reverseprimer = amplicon.reverse_primer
```

Restriction sites can be added to primers like this:

```
forwardprimer_with_BamHI = "GATC" + "GGATCC" + forwardprimer
```

### 8. PCR

```
pcr_product = pcr(forward_primer, reverse_primer, template_sequence)
```

The pcr function allows only one product to be formed. If you expect more use the Amplicon class

```
ann = Anneal(list_of_primers, template)
```

PCR products can be accessed using the products property:

```
products = ann.products
```

### 9. Ligate DNA fragments

```
seq = seq1 + seq2
```

The right end of seq1 and the left end of seq2 has to have compatible sticky ends. The resulting sequence can be circularized as shown under 5.

### 10. Primer design for assembly by Homologous recombination, Gibson assembly etc.

```
new_sequence_list = assembly_fragments(old_sequence_list)
```

The `old_sequence_list` is a list of Dseqrecord objects (or similar) and Amplicon objects. The `new_sequence_list` is a list of the same fragments where primers have been given tails to allow assembly. At least every second fragment has to be an amplicon.

### 11. Assembly by Homologous recombination, Gibson assembly etc.

```
asm = Assembly(sequence_list)
```

The assembly object can be inspected like this:

```
asm
Assembly:
Sequences.....: [33] [34] [35]
Sequences with shared homologies.: [33] [34] [35]
Homology limit (bp).....: 14
Number of overlaps.....: 3
Nodes in graph(incl. 5' & 3').....: 5
Only terminal overlaps.....: No
Circular products.....: [59]
Linear products.....: [74] [73] [73] [54] [54] [53] [15] [14] [14]
```

The report display the variables used for the assembly.

The linear and circular products can be accessed in the lists below

```
circular_product_list = asm.linear_products
```

```
linear_product_list = asm.circular_products
```