LABORATOR 8

1. Biopython

Este un set de biblioteci dedicate lucrului cu informatii, fisiere si baze de date bioinformatice. Biopython asigura mai multe cai pentru rezolvarea unei probleme.

Ofera numeroase functionalitati, precum:

- functii de parsare a formatelor utilizate in bioinformatica:
  - Blast output
  - Clustalw
  - FASTA
  - GenBank
  - PubMed si Medline
  - ExPASy files, precum Enzyme si Prosite
  - SwissProt
- functii pentru lucrul cu cele mai utilizate baze de date bioinformatice on-line:
  - NCBI – Blast, Entrez si PubMed
  - ExPASy – Swiss-Prot si Prosite
- interfete catre utilitare:
  - Blast de la NCBI
  - Clustalw program de aliniere
- o clasa standard dedicata secventelor specifice
- functii care realizeaza operatii specifice pe secvente
- algoritmi si solutii de tratare a alinierilor
- algoritmi pentru clasificari de date
- programe bazate pe GUI pentru manipularea secventelor

Instalare:
http://biopython.org/wiki/Download

Documentatie:
http://biopython.org/DIST/docs/tutorial/Tutorial.html#htoc2
http://biopython.org/DIST/docs/api/

1. Clasa Seq – obiecte de tip secventa:
   - atributul seq care contine secventa este un sir de caractere – este imutabil
   - un atribut foarte important specific secventei este alfabetul
   - se comporta similar cu stringurile, au multe functii preluate de la stringuri
   - prezinta si functii specifice care permit realizarea operatiilor pe secvente biologice (ADN, ARN, proteine)

Exemplul 1
- alfabet generic – poate fi orice tip de secventa
- secventa ambigua – poate contine si alte simboluri

```
from Bio.Seq import Seq
my_seq = Seq("AGTACACTGGT") #secventa ambigua
print my_seq
print my_seq.alphabet #alfabet generic
```

Exemplul 2 – secventa de ADN
- alfabete IUPAC
- http://biopython.org/DIST/docs/api/
- in modulul Bio.Alphabet se gasesc toate alfabetele suportate de biopython
- o secventa "unambiguous_dna" este alcatuita doar din literele A, T, C, G
- o secventa "ambiguous_dna" este alcatuita din literele A, T, C, G, plus alte simboluri; acolo unde secventierea nu a fost clara sau sigura, pot aparea si urmatoarele litere, cu urmatoarea seminificatie:

  N: orice baza
  K: G or T
  R: A or G (baza purinica)
  Y: C or T (baza pirimidinica)
  M: A or C (amino)
  S: C or G
  W: A or T

```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC
my_seq = Seq("AGTACACTGGT", IUPAC.unambiguous_dna) #secventa ADN
print my_seq
print my_seq.alphabet
```

Exemplul 3 – secventa de proteina

```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC
my_prot = Seq("AGKTRHIAK", IUPAC.protein)
print my_prot
print my_prot.alphabet
```

Exemplul 4 – secventele se comporta ca niste stringuri si sunt imutabile ca si stringurile

```
dna_seq = Seq("AGTACACTGGT", IUPAC.unambiguous_dna)
print len(dna_seq)
print dna_seq.count('A')
print dna_seq.count('AC')
print dna_seq+dna_seq
```

```
print dna_seq[2]
try:
    print my_prot + dna_seq
    #alfabete incompatibile deoarece secventele biologice sunt incompatibile
except Exception:
    print 'Nu merge'
```

Exemplul 5 – in cazul secventelor compatibile, la concatenare se face conversia la cea mai generala.

```
from Bio.Seq import Seq
from Bio.Alphabet import generic_nucleotide
from Bio.Alphabet import IUPAC
nuc_seq = Seq("GATCGATGC", generic_nucleotide)
dna_seq = Seq("ACGT", IUPAC.unambiguous_dna)
print nuc_seq.alphabet
print dna_seq.alphabet
print (nuc_seq + dna_seq).alphabet
```

Exemplul 6 – operatia de complementare a unei secvente

```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC
my_seq = Seq("GATCGATGGGCCTATATAGGATCGAAAATCGC", IUPAC.unambiguous_dna)
print my_seq
print my_seq.complement()
print my_seq.reverse_complement()

protein_seq = Seq("EVRNAK", IUPAC.protein)
try:
    print protein_seq.complement()
except ValueError:
    print 'Nu are sens operatia de complement la o proteina'

my_seq = Seq("GAUCGAUGGGCCUAUAUAGGAUCGAAAAUCGC", IUPAC.unambiguous_rna)
print my_seq
print my_seq.complement()
print my_seq.reverse_complement()
```

Exemplul 7 – Transcriptia

```
dna_seq = Seq("GATCGATGGGCCTATATAGGATCGAAAATCGC", IUPAC.unambiguous_dna)
print dna_seq
# se complementeaza secv de adn si se schimba sensul de citire cu functia
# reverse_complement()
# apoi cu functia transcribe se schimba T cu U
rna_seq = dna_seq.reverse_complement().transcribe()
print rna_seq
print rna_seq.alphabet

#reverstranscriptie
print rna_seq.back_transcribe().reverse_complement()
```

Exemplul 8 – Translatia

```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC
messenger_rna =
Seq("AUGGCCAUUGUAAUGGGCCGCUGAAAGGGUGCCCGAUAG",IUPAC.unambiguous_rna)
```

```
print messenger_rna
print messenger_rna.translate()
print messenger_rna.translate().alphabet
print messenger_rna.translate(to_stop=True) #se opreste la primul codon stop
```

Exemplul 9 – Egalitatea intre obiecte

```
seq1 = Seq("ACGT", IUPAC.unambiguous_dna)
seq2 = Seq("ACGT", IUPAC.unambiguous_dna)
print id(seq1) == id(seq2)
print str(seq1) == str(seq2)
```

2. Modulul si clasa CodonTable

Exemplul 10 – codul genetic standard

```
from Bio.Data import CodonTable
standard_table = CodonTable.standard_dna_table
print standard_table
print standard_table.start_codons
print standard_table.stop_codons
print standard_table.forward_table['TGC']
print standard_table.back_table['M']
```

3. Clasa MutableSeq – Secvente mutabile

Exemplul 11 – Secvente mutabile

```
from Bio.Seq import Seq
from Bio.Alphabet import IUPAC
my_seq = Seq("GCCATTGTAATGGGCCGCTGAAAGGGTGCCCGA", IUPAC.unambiguous_dna)
print my_seq
try:
    my_seq[5] = "G"
except Exception:
    print 'Seq este imutabila'

#crearea unei secv mutabile - metoda 1
mutable_seq = my_seq.tomutable()
mutable_seq[5] = 'G'
print mutable_seq

#crearea unei secv mutabile - metoda 2
from Bio.Seq import MutableSeq
from Bio.Alphabet import IUPAC
mutable_seq =
MutableSeq("GCCATTGTAATGGGCCGCTGAAAGGGTGCCCGA",IUPAC.unambiguous_dna)
print mutable_seq
mutable_seq[0] = 'C'
print mutable_seq

mutable_seq.remove("T")#elimina primul caracter gasit
print mutable_seq

#transformarea din secventa mutabila in secventa imutabila
new_seq = mutable_seq.toseq()
```

4. Clasa SeqRecord
   Obiectele de tip SeqRecord au urmatoarele atribute:
   - seq – obiect de tip secventa Seq
   - id – primul ID utilizat pentru identificarea secventei ("accession number")
   - name – un sir de caractere care reprezinta nume/ID secventa (in unele cazuri este acelasi cu "accesion number")
   - description – un sir de caractere care contine o descriere a secventei
   - letter_annotations - dictionar care face corespondenta intre o proprietate (cheia) si o lista/string/tuplu de aceeasi lungime cu secv (fiecarui simbol din secv i se atribuie o valoare in cadrul listei/stringului/tuplului)
   - annotations – dictionar cu diferite adnotari despre secventa
   - features – lista de obiecte de tip SeqFeature (gene, secvente codificatoare, domenii etc) cu informatii precum pozitiile genelor in genom, domeniile unei secvente proteice, etc.
   - dbxrefs - lista de referinte catre alte baze de date

Exemplul 12

```
from Bio.Seq import Seq
simple_seq = Seq("GATC", IUPAC.unambiguous_dna)
from Bio.SeqRecord import SeqRecord
simple_seq_r = SeqRecord(simple_seq)
print simple_seq_r
print simple_seq_r.seq
```

5. Clasa SeqIO
   Extragerea de informatii din fisiere standard – obiecte SeqIO pentru scrierea si citirea de secvente in/din formate specificate

Exemplul 13 – parsare FASTA

```
from Bio import SeqIO
from Bio.Alphabet import IUPAC
record = SeqIO.read(open("sequences.fasta"), "fasta")
print record
record.seq.alphabet = IUPAC.unambiguous_dna
print record
print record.dbxrefs
print record.annotations
print record.letter_annotations
print record.features
```

Exemplul 14 – parsare GenBank

```
from Bio import SeqIO
record = SeqIO.read(open("sequences.gb"), "genbank")
print record
print len(record.annotations)
print record.annotations
```

Exemplu 15 - conversie din format GenBank in FASTA

```
#metoda 1
from Bio import SeqIO
in_handle = open("sequences.gb", "r")
out_handle = open("a.fasta", "w")
records = SeqIO.parse(in_handle, "genbank")
count = SeqIO.write(records, out_handle, "fasta")
in_handle.close()
out_handle.close()
print "Converted %i records" % count

#metoda 2
from Bio import SeqIO
count = SeqIO.convert("sequences.gb", "genbank", "b.fasta", "fasta")
print "Converted %i records" % count
```

6. Clasa SeqFeature
   Extragerea de elemente caracteristice ale unei secvente din fisiere GenBank. Elementele caracteristice sunt obiecte de tip SeqFeature si au urmatoarele atribute:

- location – locatia in cadrul secventei – obiect de tip FeatureLocation care se creeaza in felul urmator:
  FeatureLocation(poz_start, poz_stop)
  Atentie: In fiserul genbank numerotarea incepe de la 1, si sunt incluse si limitele secventei, in timp ce in python secventa este numerotata de la 0 si este inclusa doar limita inferioara, iar limita superioara este exclusa.
- type – o descriere a tipului elementului caracteristic, de obicei "CDS" (Coding Sequence) sau "gene"
- ref – o referinta catre alta secventa. De exemplu genomul este retinut in mai multe regiuni (fiecare intr-un fisier). Daca o gena incepe intr-o regiune si se termina in alta, atunci campul ref va contine numarul de acces catre regiunea care contine continuarea genei
- ref_db – daca continuarea genei se afla in aceeasi baza de date, acest camp nu se completeaza, altfel se trece numele bazei de date corespunzatoare
- strand – in cazul secventelelor de ADN, indica daca gena se afla pe catena directa (are valoarea 1) sau complementara (are valoarea -1)
- qualifiers – dictionar cu informatii suplimentare despre gena
- sub_features – camp in care se retin elemente caracteristice importante ale unui element caracteristic (gena)

7. Extragerea de informatiilor din bazele de date NCBI prin intermediul sistemului Entrez

Exemplul 16 – cautarea in baze de date

```
from Bio import SeqIO
from Bio.Alphabet import IUPAC
from Bio import Entrez
Entrez.email = 'abc@gmail.com' #o adresa de e-mail

#cautarea unei gene in baza de date cu nucleotide si extragerea informatiei in
format FASTA
handle = Entrez.esearch(db="nucleotide",term="Cypripedioideae[Orgn] AND
matK[Gene]")
rec = Entrez.read(handle)
```

```
print rec["Count"]
print rec["IdList"]

#extragerea informatiei
handle = Entrez.efetch(db="nucleotide", id=rec["IdList"][0], rettype="fasta")
#salvarea informatiei extrase intr-un SeqRecord
secrec = SeqIO.read(handle, "fasta")
handle.close()

#scrierea secventei in fisierul FASTA
handle = open("search.fasta", "w")
SeqIO.write([secrec], handle, "fasta")
handle.close()
```

Cerinte:

1. Cautati in baza de date "nucleotide" gena PAX-6 si salvati pe rand informatia inregistrarilor 1, respectiv 5 in format FASTA.

2. Extrageti din baza de date "nuccore" (Genome), secventa cu id = 'NC_009084', in format FASTA. Ce tip de alfabet i se atribuie secventei?
   Schimbati tipul alfabetului in IUPAC.unambiguous_dna.
   Scrieti informatia primita intr-un fisier FASTA.

3. Extrageti din baza de date "nuccore" (Genome), secventa cu id = 'NC_016438', in format Genbank. Scrieti informatia primita intr-un fisier genbank. Utilizati urmatorii parametrii pentru functia efetch:
   ```
   Entrez.efetch(db="nuccore", id="NC_016438", rettype="gb", retmode='text')
   ```

4. Cititi o secventa dintr-un fisier GenBank intr-un obiect SeqRecord. Afisati 2 elemente ale secventei – features (gene sau CDS) si apoi extrageti si afisati tipul, locatia in secventa si catena pe care se afla acestea.

5. Cititi o secventa dintr-un fisier FASTA intr-un obiect SeqRecord si realizati urmatoarele operatii:

   a. Afisati obiectul SeqRecord.

   b. Importati urmatoarele clase:
   ```
   from Bio import SeqIO
   from Bio.Alphabet import IUPAC
   from Bio.SeqFeature import SeqFeature
   from Bio.SeqFeature import FeatureLocation
   ```

   c. Schimbati alfabetul la unambiguous_dna.

   d. Schimbati campurile "nume" si "id" ale obiectului SeqRecord, cu numarul de acces al secventei (campul nume poate avea maxim 16 caractere).

   e. Creati un obiect de tipul SeqFeature().

   f. Setati tipul obiectului creat la "gene", catena complementara si locatia genei intre nucleotida 18 si 200.

   g. Adaugati gena la lista de elemente caracteristice (features) a obiectului SeqRecord.

   h. Afisati obiectul SeqRecord. Afisati si lista de elemente caracteristice (features) din obiectul SeqRecord.

   i. Scrieti obiectul de tip SeqRecord intr-un fisier GenBank.