Corso di Laurea in Biologia Molecolare, Università di Padova
Insegnamento di Informatica e Bioinformatica

IV Esercitazione di Bioinformatica
"Python for bioinformatics"

L'obbiettivo dell'esercitazione è di familiarizzare con l'uso del linguaggio di programmazione Python in ambito bioinformatico. A partire da 10 file FASTA indipendenti, verrà creato un unico file contenente tutte le sequenze e queste verranno "lette" attraverso Python e filtrate per lunghezza.

Scaricare il file compresso contenente i dati per l'esercitazione dal seguente link:
http://compgen.bio.unipd.it/downloads/esercitazionebioinfo2015.tgz

Uso della Shell di Linux (preparazione dei dati):

Decomprimere il pacchetto nella vostra home directory (~/nomeutente); verrà creata una directory "esercitazione_bioinfo_2015"

Aprire un terminale dei comandi (cercare "terminale" tra i programmi disponibili) e spostarsi nella directory "esercitazione_bioinfo_2015"

```
cd ~/esercitazione_bioinfo_2015
```

La ~ (tilde) è carattere che indica la home directory dell'utente (/home/username)
Dare un'occhiata ai file contenuti nella directory:

```
ls
```

Verranno visualizzati i nomi dei file presenti nella directory, tra cui 10 sequenze in formato FASTA di trascritti riconducibili a geni coinvolti nel cancro al seno. Ogni sequenza è in un file separato.

Concateniamo tutte le sequenze FASTA in un unico file. NB: ogni sequenza ha una propria intestazione, quindi concatenandole in un unico file saranno comunque riconoscibili.

```
cat seq*.fasta > all.fasta
```

* è un carattere speciale (*wild card*) che corrisponde ad uno o più caratteri.
Il simbolo > redireziona l'output di un comando (cat in questo caso) in un file.

Ispezionare il file appena creato con:

```
less all.fasta
```

Verificare che il numero di sequenze nel file creato sia corretto:

```
grep ">" all.fasta
```

e contare quante righe vengono date in output, oppure per farlo in maniera automatica usare l'opzione -c del comando grep:

```
grep -c ">" all.fasta
```

Se vogliamo ottenere informazioni sul comando grep:

```
man grep
grep -h
```

Programmazione in Python (processamento dei dati)

Dal terminale avviare l'interprete interattivo di Python (`python, ipython, idle`).

Il codice lo trovate nel file `esercitazione.py`

Importare la libreria Biopython (http://biopython.org) contenente routines di tipo bioinformatico utili alla manipolazione di file FASTA.
Biopython non è una libreria standard di Python e se non presente deve essere installata separatamente. La libreria è già installata nelle macchine del laboratorio.
Caricheremo solo un modulo (SeqIO) della libreria che utilizzeremo per maneggiare i file FASTA.

```
>>> from Bio import SeqIO
```

Maggiori informazioni sul modulo SeqIO: http://biopython.org/wiki/SeqIO e http://biopython.org/DIST/docs/api/Bio.SeqIO-module.html

Apriamo il file contenente tutte le sequenze:

```
>>> fileHandler = open("all.fasta")
```

Leggiamo le sequenze in formato FASTA utilizzando il metodo to_dict del modulo SeqIO

```
>>> recordDict = SeqIO.to_dict(SeqIO.parse(fileHandler, "fasta"))
```

Visualizziamo le chiavi del dizionario (i nomi delle sequenze fasta):

```
>>> recordDict.keys()
```

Visualizziamo il trascritto con UCSC ID uc021pvw.1

```
>>> seqObject = recordDict.get("uc021pvw.1")
```

Visualizziamo la sequenza nucleotidica accedendo all'attributo *seq* dell'oggetto seqObject:

```
>>> print seqObject.seq
```

Visualizziamo l'identificativo del trascritto:

```
>>> print seqObject.name
```

Qual'è la lunghezza del trascritto uc021pvw.1?

```
>>> len(seqObject)
>>> len(seqObject.seq)
```

Quali sono I primi 100 nucleotidi del trascritto uc021pvw.1?

```
>>> str(seqObject.seq[0:100])
```

Gli oggetti Bio.Seq.Seq (attributo seq di SeqObject) si possono processare attraverso l'operatore di slicing come le stringhe, le liste etc...
Quello che otteniamo è ancora un oggetto Bio.Seq.Seq e quindi per avere una stringa facciamo il cast esplicito con la funzione str

Vogliamo ora calcolare la lunghezza media delle nostre sequenze. La variabile `lenghtList` (una lista in python) conterrà le lunghezze delle sequenze considerate.

```
>>> lenghtList = []

>>> for record in recordDict.keys():
...   length = len(recordDict[record])
...   lenghtList.append(length)
```

Visualiziamo la lista contenente le lunghezze dei trascritti

```
>>> lenghtList
```

Da quanti nucleotidi è composto il trascritto più lungo?

```
>>> max(lenghtList)
```

...e quello più corto?

```
>>> min(lenghtList)
```

Calcoliamo ora la media delle lunghezze dei trascritti

```
>>> 1.0 * sum(lenghtList) / len(lenghtList)
```

Filtriamo ora le sequenze più corte di 1000 nucleotidi

```
>>> filteredSequences = []
>>> for record in recordDict.keys():
...   sequence = recordDict[record]
...   if len(sequence) > 1000:
...     filteredSequences.append(sequence)

>>> len(filteredSequences)
```

Apriamo un file in scrittura (“w”) dove salveremo i trascritti più lunghi di 1000 basi

```
>>> outputFile = open("my_sequences.fasta", "w")
```

Scriviamo le sequenze in formato FASTA

```
>>> SeqIO.write(filteredSequences, outputFile, "fasta")
>>> outputFile.close()
```