Stefan Kurtz **Universität Hamburg**
Mathias von Behren **Zentrum für Bioinformatik**

Programmierung in der Bioinformatik
Wintersemester 2015
Übungen zur Vorlesung: Ausgabe am 11.01.2016

Punktevergabe:

- Aufgabe 12.1: 5 Punkte
- Aufgabe 12.2: 5 Punkte

Aufgabe 12.1 Sie haben in früheren Aufgaben Iteratoren wie `each` oder `each_line` auf geeignete Objekte angewendet, z.B. wie folgt:

```
my_array.each do |element|
  puts element
end
```

Dabei wird der Methode `each` der Block **`do`** ... **`end`** übergeben und mit jedem einzelnen Element des Arrays aufgerufen.

In dieser Aufgabe geht es darum einen Iterator für Multi-Fasta-Dateien zu schreiben. Dazu sollten Sie sich den Abschnitt über Iteratoren im Skript ansehen. Schreiben Sie nun eine Klasse, deren Instanzen mit einem Dateinamen initialisiert werden. Implementieren Sie für diese Klasse eine `each`-Methode, in der **`yield`** auf den aktuellen Header `header` und die entsprechende Sequenz `sequence` angewendet wird. Bei Verwendung von `each` muss ein Block mit den Parametern `header` und `sequence` übergeben wird:

```
my_fasta = FastaIterator.new("filename.fasta")
my_fasta.each do |header, sequence|
  #prozessiere header und sequence
end
```

Schreiben Sie ein Skript, das diese Klasse und die Methode `each` verwendet, um folgendes zu tun:

- Formatierte Ausgabe der Fasta-Einträge, wobei jeweils höchstens 70 Zeichen für jede Zeile mit Sequenzinformation verwendet wird.
- Berechnung der Längenverteilung der Sequenzen in 100er Bins (1-100,101-200,201-300,...). D.h. es wird für alle $i \geq 0$ die Anzahl der Sequenzen der Länge ℓ mit $1+100i \leq \ell \leq 100(i+1)$ ausgegeben. Dabei erscheinen in der Ausgabe nur positive Anzahlen. Beispiel: Für die in STiNE bereitgestellte Datei `swiss.fna` soll die Ausgabe der Längenverteilung wie folgt aussehen:

```
1-100: 2
101-200: 2
201-300: 5
301-400: 1
401-500: 2
901-1000: 1
1301-1400: 1
```

Aufgabe 12.2 Erstellen Sie ein Ruby-Skript, das für alle Einträge in einer Multi-Fasta Datei die Übersetzung des längsten offenen Leserahmens (ORF) in eine Proteinsequenz ausgibt. Dabei soll der Header wieder mit ausgegeben werden, so dass Ihre Ausgabe ebenfalls der Fasta-Konvention folgt.

Zur Erinnerung:

- Jeder Eintrag in einer Multi-Fasta Datei besteht aus folgenden Komponenten: Einer mit „>“ beginnenden Header-Zeile und der Sequenz selbst, die sich über mehrere Zeilen erstrecken kann und mit der nächsten Header-Zeile oder dem Dateiende aufhört.
- Ein ORF ist eine Basenfolge, die mit einem Start-Codon (`atg`) beginnt, und im selben Leserahmen mit einem Stop-Codon (`taa|tag|tga`) endet (Siehe Aufgabe 10.2 zur Markierung von ORFs).

Am einfachsten ist es, die Klasse `FastaIterator` aus Aufgabe 12.1 um eine passende Methoden zu erweitern.

Bedenken Sie beim Identifizieren des längsten ORFs, dass DNA aus zwei Strängen mit jeweils drei Leserahmen besteht, die jeweils auf Start-/Stopcodons untersucht werden müssen. Sie müssen also zum Untersuchen des Gegenstranges die Sequenz umkehren und das Komplement bilden, also `ACGT` entsprechend der DNA-Bindungsregeln ersetzen (`A`↔`T`, `C`↔`G`).

In Stine finden Sie eine Datei, die Ihnen beim Übersetzen der Basentriplets in Aminosäuren hilft. Eine Beispieldatei `seq-with-orfs.fna` mit DNA-Sequenzen und die Datei `seq-with-orf-trans.fna` mit der erwarteten Ausgabe finden Sie in Stine. Bitte stellen Sie sicher, dass die Ausgabe Ihres Skriptes genau der erwarteten Ausgabe entspricht.

Die Lösungen zu diesen Aufgaben werden am 25.01.2016 besprochen.