Institut für Informatik
Praktische Informatik und Bioinformatik
Prof. Dr. Ralf Zimmer

Wintersemester 2019/20

Übungen zum Propädeutikum Programmierung in der Bioinformatik

Blatt 5

Termin: Dienstag, 19. November 2019

Übung 1 *Grundlegende `DnaSequence`-Klasse*

In den bisherigen Übungsblättern war in einigen Aufgaben verlangt mit DNA-Sequenzen zu arbeiten. Schreibe jetzt eine eigene Klasse in der eine DNA-Sequenz modelliert wird:

1. Erstelle eine neue Klasse `DnaSequence`. Diese soll folgende Attribute haben:
   (a) `String sequence` (eigentliche DNA-Sequenz)
   (b) `String id` (Identifier der Sequenz, z.B. aus dem FASTA-Header)
2. Schreibe einen Konstruktor `DnaSequence(String seq, String id)` der diese Variablen belegt und ein `DnaSequence`-Objekt erzeugt.
3. Schreibe zwei Methoden `getSequence()` und `getId()`, welche die Strings in `sequence` bzw. `id` zurückgeben. Schreibe zusätzlich eine Methode `setId(String s)` welche die momentane ID des Objekts mit `s` ersetzt.
4. Erstelle eine neue Klasse `Runner`. Diese dient nur zum Testen der Klasse `DnaSequence` und modelliert selbst nichts. Lege dort eine `main`-Methode an, und erstelle in dieser zwei (oder mehr) beliebig befüllte `DnaSequence`-Objekte.
   Gib mit Hilfe der `getSequence`- und `getId`-Methoden die Attribute der beiden Objekte auf dem Terminal aus. Ändere dann eine ID mit `setId` und teste, ob es erfolgreich war. Die folgenden Aufgaben beziehen sich wieder auf `DnaSequence`. Teste diese dann auch wieder in der `main`-Methode der Klasse `Runner`.
5. Schreibe eine Methode `length()`, die die Länge der Sequenz zurückgibt.
6. Schreibe eine Methode `isValidSequence()`, die überprüft ob die Sequenz eines `DnaSequence`-Objekt auch wirklich nur aus den gültigen Buchstaben `A`, `G`, `C` und `T` besteht. Falls die Sequenz gültig ist, soll die Methode `true` zurückgeben, andernfalls `false`.
7. Verwende nun die gerade geschriebene Methode `isValidSequence()` um im Konstruktor zu testen, ob die dem Konstruktor übergebene Sequenz auch gültig ist. Falls die übergebene Sequenz ungültig ist, soll beim Aufruf des Konstruktors eine kurze Warnung auf dem Terminal ausgegeben werden.

Lösung 1

```java
public class DnaSequence {
    String id;
    String sequence;

    public String getId() {
        return id;
    }

    public String getSequence() {
        return sequence;
    }

    public void setId(String id) {
        this.id = id;
    }

    public DnaSequence(String id, String sequence) {
        this.id = id;
        this.sequence = sequence;
        if (!this.isValidSequence()) System.out.println("Achtung: Keine gültige DNA Sequenz!");
    }

    public int length() {
        return this.sequence.length();
    }

    public boolean isValidSequence() {
        for (char c : sequence.toCharArray()) {
            if (c != 'A' && c != 'G' && c != 'C' && c != 'T') return false;
        }
        return true;
    }

}
```

```java
public class Runner {
    public static void main(String[] args) {
        DnaSequence dnaseq1 = new DnaSequence("EINE_ID", "GTATCXCATTTG");
        DnaSequence dnaseq2 = new DnaSequence("ANDERE_ID", "GATTACA");
    }
}
```

Übung 2 *Rosalind: Reverse Complement*

Die DNA-Doppelhelix besteht aus zwei gegenläufigen Strängen (jeweils mit 5'- und 3'-Ende), die durch komplementäre Basenpaare verbunden sind, wie hier skizziert:

$$\begin{array}{c} \longrightarrow \\ \texttt{5'···AAAACCCGGT···3'} \\ \texttt{3'···TTTTGGGCCA···5'} \\ \longleftarrow \end{array}$$

Was wäre nun das Reverse Complement zur oberen Sequenz `AAAACCCGGT`? Diese Sequenz verläuft 5'→3'. Das Reverse Complement ist dann die Sequenz auf dem (unteren) Komplementärstrang, allerdings nicht wie in der Skizze von links nach rechts (also 3'→5') verlaufend, sondern wie der obere Strang 5'→3': `ACCGGGTTTT`.

In dieser Aufgabe soll die Klasse `DnaSequence` erweitert werden. Schreibe eine Methode `reverseComplement()`, welche das Reverse Complement eines `DnaSequence`-Objekts berechnet und als *eigenständiges* `DnaSequence`-Objekt zurückgibt. Die Korrektheit deiner Lösung kannst du auf Rosalind: Complementing a Strand of DNA überprüfen.

Lösung 2

```java
public static String staticReverseComplement(String dna) {
    int length = dna.length();
    char[] reverseComplement = new char[length];
    for (int i = 0; i < length; i++) {
        if (dna.charAt(i) == 'A') reverseComplement[length-(i+1)] = 'T';
        else if (dna.charAt(i) == 'T') reverseComplement[length-(i+1)] = 'A';
        else if (dna.charAt(i) == 'G') reverseComplement[length-(i+1)] = 'C';
        else if (dna.charAt(i) == 'C') reverseComplement[length-(i+1)] = 'G';
    }
    return new String(reverseComplement);
}

public DnaSequence reverseComplement() {
    return new DnaSequence("reverse_" + this.id, staticReverseComplement(this.sequence));
}
```