

Übungen zum Bioinformatik-Tutorium

Blatt 6

Termin: Dienstag, 27.11.2018, 11 Uhr

1. Klassen und Methoden

Diese Aufgabe wird auf den folgenden Übungsblättern schrittweise erweitert. Achte also darauf, sorgfältig zu arbeiten und Probleme rechtzeitig zu klären.

- (a) Erzeuge eine neue Klasse **Protein**. Diese Klasse soll folgende **private** Felder enthalten:
 - (i) **String id** (Datenbank-Identifizier)
 - (ii) **String sequence** (Aminosäuresequenz des Proteins)
- (b) Erzeuge einen Konstruktor der diese Instanzvariablen belegt. Als Parameter erhält er also die Id und Sequenz.
- (c) Erstelle für beide Instanzvariablen Getter und Setter.
- (d) Schreibe zusätzlich eine Methode **public int length()**, die die Länge der Sequenz zurückgibt.
- (e) Schreibe eine statische Methode, die für einen String überprüft ob er eine Proteinsequenz ist, also nur aus Buchstaben, die für eine der 21 Aminosäuren kodieren besteht. Warum macht es Sinn, dass diese Methode statisch ist? Überprüfe nun auch im Konstruktor ob die übergebene Sequenz eine gültige Aminosäuresequenz ist und gib sonst eine Fehlermeldung aus.
- (f) Schreibe eine weitere Klasse **Runner** die eine **main** Methode enthält. In dieser **main** Methode sollen drei Instanzen der Klasse **Protein** angelegt werden. Gib jeweils die Id und Länge der Sequenz auf stdout aus.

2. Binärbäume

Auch diese Aufgabe wird in den folgenden Übungsblättern erweitert. Sei folgendes Gerüst einer Klasse für die Knoten eines Binärbaumes gegeben:

```
public class BinaryNode {
    private BinaryNode left;
    private BinaryNode right;
}
```

Erweitere die Klasse um folgende Instanzvariablen/Methoden/Konstruktoren, so dass die main-Methode unten (in der Klasse BinaryTreeTest) ausführbar ist.

- (a) Instanzvariable `private String name`
- (b) Getter für `left`, `right` und `name`
- (c) Methode `String getNamesDfs()` (siehe unten)
- (d) Konstruktoren für innere Knoten und Blätter

Die Methode `getNamesDfs` soll die Knoten-Namen in *depth-first* Sortierung konkatenieren. Für das Beispiel, das in `BinaryTreeTest` erzeugt wird, muss die Ausgabe wie folgt sein:

```
(5; (4; (1; null; null); (2; null; null)); (3; null; null))
```

```
public class BinaryTreeTest {
    public static void main(String[] args) {
        BinaryNode n1 = new BinaryNode("1");
        BinaryNode n2 = new BinaryNode("2");
        BinaryNode n3 = new BinaryNode("3");
        BinaryNode n4 = new BinaryNode("4",n1,n2);
        BinaryNode n5 = new BinaryNode("5",n4,n3);
        System.out.println(n5.getNamesDfs());
    }
}
```