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

Wintersemester 2018/19

Übungen zum Bioinformatik-Tutorium

Blatt 11

Termin: Dienstag, 15.01.2019, 11 Uhr

1. Collections

(a) In Aufgabe 1 auf Blatt 10 wurden die Klassen Person, Student und Professor implementiert. Lies die Datei "/home/proj/tutorium_bioinformatik/studenten.txt" ein und erzeuge die entsprechenden Objekte. Vorname und Nachname sollen einfach durch ein Leerzeichen zusammengefügt werden. Die Objekte sollen in einer `ArrayList` gespeichert werden. Gib die enthaltenen Objekte aus.

(b) Verwende `Collections.sort()` um diese ArrayList nach dem Namen lexikografisch zu sortieren. Führe die notwendigen Veränderungen in der Klasse `Person` durch, um dieser Klasse eine natürliche Ordnung bezüglich des Namens zu geben. Gib abermals das Ergebnis aus.

(c) Soll nach weiteren Eigenschaften (also nicht der natürlichen) sortiert werden, bietet das `Collections` framework die Klasse `Comparator` an. Implementiere einen `Comparator<Person>`, der nach dem Alter einer Person sortiert. Teste deine Implementierung und gib dein Ergebnis aus.

(d) Verwende den `Iterator` der ArrayList, um der Reihe nach in einer while-Schleife auf alle Objekte des Vektors zuzugreifen und sie zeilenweise auszugeben. Verwende ebenfalls die analoge for-each-Schleife.

(e) Benutze `Collections.binarySearch()` um einen Studenen zu finden, der 18 Jahre alt ist und gib den gefundenen Studenten aus.

(f) Informiere dich in der Dokumentation, was zurückgegeben wird, wenn das gesuchte Objekt nicht vorhanden ist. Erweitere deine Ausgabe, sodass in diesem Fall der nächstältere Student ausgegeben wird.

2. Studentenverwaltung

(a) Schreibe eine Klasse `StudentenVerwaltung` mit den folgenden 5 Feldern

(i) `List<Student> student_list`

(ii) `Set<Student> student_set`

(iii) `SortedSet<Student> student_set_sorted`

(iv) `Map<Integer, Student> matrikelnummer2student`

(v) `Map<Integer, Set<Student>> age2students`

(b) Schreibe einen Konstruktor, der die Datei "/home/proj/tutorium_bioinformatik/vieleLeute.txt" einliest und die Collections sinnvoll befüllt.
Hinweis 1: bei den Maps ist zu beachten, dass eine Matrikelnummer immer einen eindeutigen Studenten referenziert, während es mehrere Studenten des selben Alters geben kann.
Hinweis 2: Java bietet genau eine Klasse, die das Interface `SortedSet` implementiert und zwei Klassen, die `Set` implementieren. verwende für `student_set` nicht die sortierte Variante.

(c) Damit Objekte in einer HashMap korrekt funktionieren, muss die entsprechende Klasse die Methoden `hashCode()` und `equals()` implementieren. Implementiere in `Student` die fehlende Methode `hashCode()`.
Hinweis: der Hashcode für einen Integer ist in der Regel die Integerzahl selbst.

(d) Implementiere die Methoden `public int getNumberOfStudents()` und `public int getNumberOfUniqueStudents()`. Erstere soll die Anzahl der eingelesenen Studenten ausgeben (also der Zeilenzahl der Eingabedatei entsprechen), letztere soll die Zahl der eindeutigen Studenten ausgeben (es gibt Duplikate). Was passiert, wenn `hashCode()` nicht implementiert ist?

(e) Implementiere die Methoden `public Student getByMatrikel(int nummer)` und `public Set<Student> getByAge(int age)`. Gib den Studenten mit Matrikelnummer 826707947 aus, sowie alle Studenten, die 18 Jahre alt sind.