paluno - The Ruhr Institute for Software Technology
Prof. Dr. Michael Goedicke

Miniprojekt 3

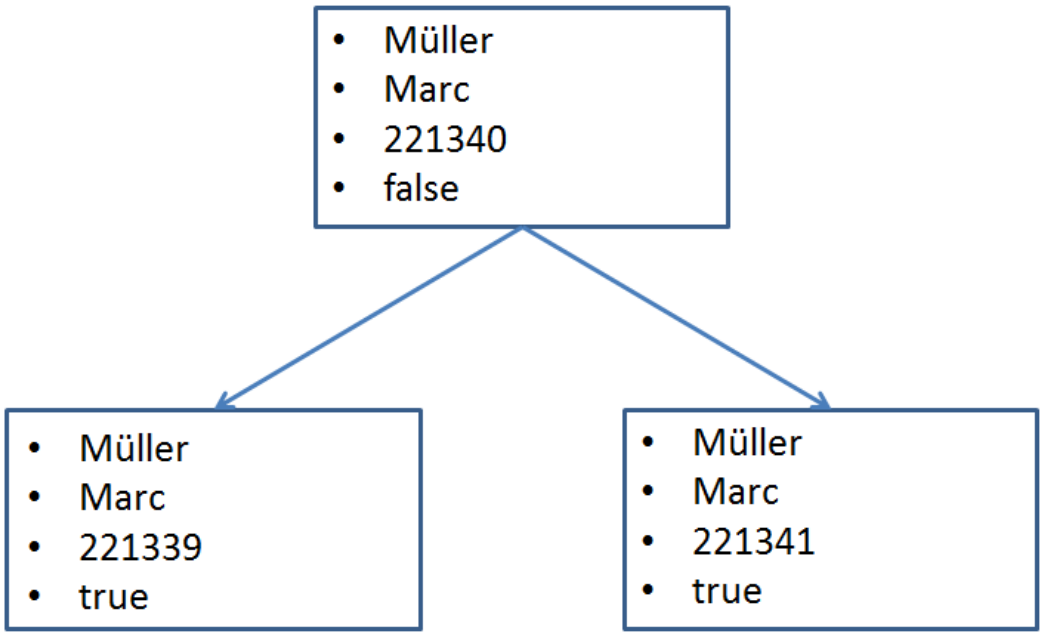
Abgabefrist: Dienstag, 27. November 2018, 23:59 Uhr

In diesem Miniprojekt befinden sich zwei Klassen, `Phonebook` und `Person`. Die Klasse `Person` deklariert einen String `lastName` für den Nachnamen der Person, einen String `firstName` für den Vornamen der Person, einen Integer `number` für die Telefonnummer der Person und einen boolean `married`, der angibt, ob die Person verheiratet ist. Sie verweist zudem auf ein Feld `leftSuccessor` und ein Feld `rightSuccessor` vom Typ `Person`. Die Klasse `Phonebook` deklariert ein Feld `firstEntry` vom Typ `Person`, das die Wurzel eines binären Baums darstellen soll. Alle Elemente im Baum sind vom Typ `Person`. Die Klasse `Phonebook` stellt außerdem eine `main`-Methode zur Verfügung.

Wie in einem richtigen Telefonbuch sollen die Einträge in unserem Baum stets folgendermaßen sortiert sein:

1. Nachname der Person
2. Bei Gleichheit des Nachnamens: Vorname der Person
3. Bei Gleichheit von Vor- und Nachname: Telefonnummer der Person.

In einem Binärbaum bedeutet dies: Der linke Nachfolger eines Eintrags ist stets kleiner im Sinne des Sortierkriteriums als der Elternknoten und der rechte Nachfolger ist stets größer im Sinne des Sortierkriteriums als der Elternknoten. Ein minimales Beispiel befindet sich in folgendem Bild:



Dabei ist der Eintrag für Marc Müller mit der Telefonnummer 221339 der leftSuccessor des Eintrages mit der Telefonnummer 221340 und der Eintrag mit der Telefonnummer 221341 der rightSuccessor des Eintrags mit der Telefonnummer 221340.

Die vorhandenen Klassen-, Variablen- und Methodennamen dürfen nicht verändert werden! Es ist dagegen erlaubt, bei Bedarf weitere Methoden zu ergänzen oder den Inhalt der `main`-Methode für eigene Tests zu verändern. Für alle Aufgaben sind zwei Methoden vorgegeben, je eine mit `public` und eine mit `private` deklariert. Die mit `public` deklarierte Methode sollte die mit `private` deklarierte aufrufen, in der dann die Rekursion stattfindet. Sie können die Methoden mit dem Modifier `private` beliebig anpassen und sogar Rückgabe und Parametrisierung ändern, wenn Sie das der Lösung näher bringt. Sie können davon ausgehen, dass jede Person stets einen eigenen Telefonanschluss besitzt, d.h. es gibt zu jeder Telefonnummer höchstens eine Person, welcher diese Telefonnummer zugewiesen ist.

Bitte beachten Sie, dass für die jeweilige Testatzulassung die Abgabe einer (nicht notwendigerweise korrekten) Lösung zum Miniprojekt erforderlich ist!

Aufgabe 1: Einfügen eines Eintrags

Implementieren Sie die Methode `insertPerson(String lastName, String firstName, int number)`, die eine Person mit den gegebenen Parametern erzeugt und an der richtigen Stelle in den Baum einsortiert. Sie können die Methode `compareTwoPersons(Person person1, Person person2)` benutzen, um zwei Personen im Sinne des Sortierkriteriums miteinander zu vergleichen.

Aufgabe 2: Finden eines Eintrags

Implementieren Sie die Methode `findPerson(String lastName, String firstName, int number)`, die entscheiden soll, ob eine Person mit den gegebenen Parametern im Telefonbuch zu finden ist.

Aufgabe 3: Zählen aller Einträge

Implementieren Sie die Funktion `count()`, sodass diese die Anzahl aller Einträge des Telefonbuchs zurückgibt.

Aufgabe 4: Suchen von Personen

Implementieren Sie die Methode `findAllPersonsByFirstName(String firstName)` in der Klasse `PhoneBook`, die ein Array erstellt, welches alle Personen mit dem übergebenen Vornamen beinhaltet. Das Array soll zudem nach den Sortierkriterien des Telefonbuchs sortiert sein, d.h. die Person „Marc Müller“ mit der Telefonnummer „221339“ muss im Array vor der Person „Marc Müller“ mit der Telefonnummer „221340“ zu finden sein. Wenn es keine Einträge zu dem Vornamen gibt, dann soll ein leeres Array zurückgegeben werden. Die Methode soll also nie `null` zurückgeben.

Aufgabe 5: Ändern eines Eintrags

Aufgrund einer Gesetzesänderung müssen alle Personen mit dem Nachnamen „Hochzeit“ auch tatsächlich verheiratet sein. Implementieren Sie die Methode `marryTheHochzeits()` so, dass diese bei allen Personen mit dem Nachnamen „Hochzeit“ den Wert von `married` auf `true` setzt.

Aufgabe 6: Löschen von Einträgen

Natürlich muss es auch möglich sein, Einträge aus dem Telefonbuch zu löschen. Implementieren Sie die Methode removePersonFromPhoneBook(String lastName, String firstName, int number), die genau dies tut. Beachten Sie dabei, dass beim Löschen der rechte und linke Nachfolger des zu löschenden Eintrags im Baum verbleiben und korrekt in diesen einsortiert werden müssen.

Aufgabe 7: Eintraege ändern und neu einsortieren

Wenn Menschen heiraten, dann ändert manchmal zumindest einer der beiden Ehepartner seinen Nachnamen. Implementieren Sie die Methode `changePerson(String lastName, String firstName, int number, String newLastName)`, die dafür sorgt, dass die Person mit den gegebenen Parametern umbenannt und anschließend korrekt in den Baum einsortiert wird.