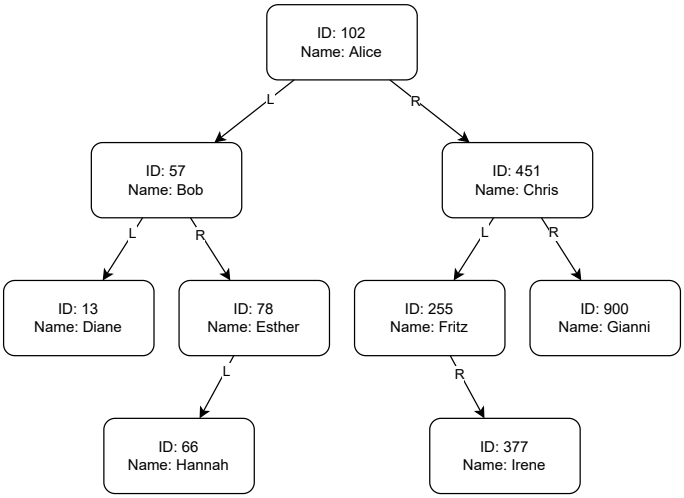
4 Datenstrukturen

Ein Binärbaum ist eine Datenstruktur, welche die Suche von Daten beschleunigt. Ein Binärbaum besteht aus Knoten, die Daten speichern können. Jeder Knoten kann einen linken und einen rechten Kindknoten besitzen. Der linke Kindknoten enthält einen Wert der kleiner oder gleich dem Wert des Wurzelknotens ist, und der rechte Kindknoten enthält einen Wert der größer ist.

Im folgenden sollen Sie eine Personendatenbank schreiben, die mit einem Binärbaum verwaltet wird. Jeder Person wird eine zufällige ID zwischen 1 und 1000 zugeordnet, welche im Binärbaum zur Suche verwendet werden soll. Der entstehende Binärbaum kann beispielsweise wie folgt aussehen:



Aufgaben

1. Erstellen Sie ein Projekt und initialisieren Sie es mit einer Git-Repository. Legen Sie zu jedem der folgenden Aufgaben einen Commit an sobald Sie diese gelöst haben.
2. Legen Sie die Klasse `Person` mit den Feldern *Name* und einem Integer *ID* an.
3. Erstellen Sie einen Binärbaum, in dem Personen hinterlegt werden können. Nachdem eine Person dem Binärbaum hinzugefügt wurde, soll diese mittels der ID wieder zurückgegeben werden können. Legen Sie hierfür die Klasse `Binaerbaum` an, welche die Methoden *hinzufuegen(p: Person): void* und *finden(id: int): Person* besitzt.

   Hinweis: Binärbäume sind rekursiv! Die beiden Kindknoten einer Wurzel sind wiederum selber die Wurzeln eines Binärbaumes.
4. Ändern Sie Ihre Klasse für Binärbäume so ab, dass beliebige Typen gespeichert werden können, solange diese das Interface `Comparable` implementieren. Nutzen Sie dafür Javas generische Typisierung.

Tipp

Keine Angst Code zu löschen! Mit Git kann immer zu dem Stand eines Commits zurückgesprungen werden.