ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Dr. Jochen Hoenicke
Alexander Nutz

14.7.2014
Abgabe: Montag 21.7.2014
via Daphne/Subversion

11. Übungsblatt zur Vorlesung
Einführung in die Informatik

Aufgabe 1: Testen
Schreiben Sie eine Methode, die auf Eingabe zweier Mengen A, B von Ganzzahlen deren Kreuzprodukt $A \times B$ berechnet.
Das Kreuzprodukt zweier Mengen A, B ist die Menge aller Paare deren erster Eintrag aus A und deren zweiter Eintrag aus B stammt.
Beispiel: $A = \{1, 2, 3\}, B = \{3, 4\}$, dann ist $A \times B = \{(1, 3), (2, 3), (3, 3), (1, 4), (2, 4), (3, 4)\}$.
Beginnen Sie die Entwicklung indem sie einige Tests schreiben, die möglichst gut abbilden, was das Program tun soll. Implementieren Sie anschließend die Methode und führen Sie die Tests darauf aus um zu überprüfen, ob Ihre Implementierung die erwarteten Ergebnisse liefert.
Hinweis: Zur Repräsentation von Mengen eignet sich die Klasse `HashSet`, für die Paare können Sie entweder eine eigene Klasse bereitstellen, oder einfach zweielementige Arrays oder ArrayLists benutzen.

Aufgabe 2: Debuggen und Testen
Die Fibonacci-Folge ist definiert durch $F(0) = 0, F(1) = 1, F(n) = F(n-1)+F(n-2)$. Die rekursive Implementierung, die sich direkt aus dieser Definition ergibt, ist sehr ineffizient (ca. 2^n Methodenaufrufe). Wesentlich schneller und speichereffizienter ist die iterative Variante (ca. n Schleifendurchläufe). Diese berechntet $F(n)$, indem sie mit $F(0)$ und $F(1)$ beginnt und in einer Schleife immer die nächste Fibonacci-Zahl aus den zwei vorherigen berechnet. Dazu muss das Programm nur drei Variablen besitzen.
Stellen Sie sich vor, Sie wollen eine iterative Lösung zur Berechnung der n-ten Fibonacci-Zahl implementieren. Sie sind schon recht nah dran, Ihr Stand ist der von `Fibo.java` auf der Vorlesungswebsite.
Die gelieferten Ergebnisse, sehen aber noch nicht nach den Fibonacci-Zahlen aus. Schreiben Sie einige Testfälle. Die Klasse `FiboTest` auf der Vorlesungswebsite soll ihnen dabei die Arbeit erleichtern.
Versuchen Sie anschließend zu verstehen, was schief läuft, indem Sie an strategischen Punkten Debug-Anweisungen platzieren.
Reparieren Sie das Programm.