

Übungsblatt 8: Rekursion

Speichern Sie die Programme im Paket `de.hska.inf01.rekursion` ab.

Aufgabe 1 (Lineare Rekursion)

Implementieren Sie eine rekursive Methode, die in einem Feld vom Typ `int` das kleinste Element heraussucht und zurückgibt. Das zu durchsuchende Feld soll als Parameter Ihrer Methode übergeben werden.

Die Lösung soll objekt-orientiert sein (keine statischen Methoden). Die Rekursion darf bei Ihrer Lösung nicht verzweigen: ein Aufruf Ihrer Methode darf höchstens einen weiteren rekursiven Aufruf zur Folge haben.

Überprüfen Sie Ihre Implementierung mit einer JUnit-Testklasse. Beachten Sie auch besonders extreme Randbedingungen bei Ihren Testfällen wie zum Beispiel Aufrufe der Methode mit `null`, einem Feld der Länge 0 oder einem Feld der Länge 1 000 000. Überlegen Sie kurz, wie sie die Methode iterativ hätten implementieren können (sie müssen Sie nicht dazu implementieren), und vergleichen Sie diesen Lösungsansatz mit der rekursiven Variante. Welche Implementierungsvariante würden Sie für die Praxis wählen?

Aufgabe 2 (Verzweigende Rekursion)

Implementieren Sie eine rekursive Methode `public long getBinomialKoeffizient(int n, int k)`, die den Binomialkoeffizienten mit Hilfe des Pascalschen Dreiecks berechnet:

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k} \text{ und } \binom{n}{0} = \binom{n}{n} = 1$$

Ihre Implementierung der Methode darf keinerlei Felder zur Speicherung von Zwischenergebnissen oder `for/while` Schleifen zur Berechnung von (Teil-)Ergebnissen verwenden. Die Lösung soll objekt-orientiert sein (keine statischen Methoden). Testen Sie Ihre Implementierung mit Hilfe einer JUnit Testklasse. Überlegen Sie sich auch, wie sich ihre Methode bei Aufruf von negativen Zahlen oder Zahlen mit $k > n$ verhalten soll und wie sie dieses Verhalten mit Testfällen überprüfen.

Erweitern Sie Ihre Implementierung, so dass die Anzahl aller rekursiver Aufrufe für einen Aufruf von `getBinomialKoeffizient()` gezählt wird und mit einer Methode zurückgegeben bzw. wieder auf 0 gesetzt werden kann. Implementieren Sie eine weitere Klasse mit einer `main`-Routine, wo Sie für alle i von 0 bis 30 die Binomialkoeffizienten $\binom{i}{\frac{i}{2}}$ berechnen und für jeden Wert von i die zugehörige Anzahl der rekursiven Aufrufe für die Berechnung des Binomialkoeffizienten auf dem Bildschirm ausgeben. Wie lange dauert die Berechnung ungefähr für $i = 30$, wie lange wird wohl die Berechnung für $i = 35$ und $i = 40$ dauern?

Vergleichen Sie die rekursive Implementierung mit der Implementierung aus der Vorlesung. Welche Vor- und Nachteile haben beide Lösungen?