**Fachbereich Informatik**
Dr. Michael Haupt
Visar Januzaj
Johannes Kinder
*Sommersemester 2010*

TECHNISCHE UNIVERSITÄT DARMSTADT

# Grundlagen der Informatik 2

## Praktikumsaufgabe 1

Abgabe: 24.05.2010, 12:00

**Rucksackproblem.** Gegeben sind $n$ Objekte $O_0, \ldots, O_{n-1}$. Jedes Objekt $O_i$ hat ein Gewicht $w(O_i)$ und einen Wert $v(O_i)$. Mit diesen Objekten soll ein Rucksack gefüllt werden. Der Rucksack hat die Kapazität $C$, d.h. die Summe der Gewichte der Objekte, die sich im Rucksack befinden, darf $C$ nicht übersteigen.

Füllen Sie den Rucksack so, daß die Summe der Werte aller Objekte im Rucksack maximal ist. **Achtung:** Jedes Objekt darf höchstens einmal in den Rucksack gepackt werden.

**Hinweise.** Import der Quelltext-Vorgaben: Laden Sie die Quelltext-Vorgaben für Praktikumsaufgabe 1 aus dem moodle-Portal herunter (Project1.zip).

Importieren Sie das Projekt in Eclipse (File, Import, General, Existing Projects Into Workspace, Select Archive File, Browse, Project1.zip auswählen, Finish).

**Zu implementieren.** Wie immer sind die zu bearbeitenden Stellen im Code mit **TODO:** markiert. Im einzelnen:

1. Implementieren Sie für die Klasse `Rucksack` in `Rucksack.java` den Konstruktor, die verschiedenen Zugriffsmethoden für Kapazität, Gewicht und Wert, die Methoden `putObject` und `removeObject`, die ein Objekt vom Typ `PackingObject` (in der Vorgabe bereits implementiert) hinzufügen bzw. entfernen und ein entsprechendes `contains`.

   Implementieren Sie außerdem die Methode `copyFrom`, die einen anderen Rucksack vollständig kopiert. Diese Methode können Sie nutzen, um sich in den Algorithmen den jeweils aktuell besten gefundenen Rucksack zu merken.

2. Implementieren Sie eine rekursive Tiefensuche `solveRecursive` in `RucksackProblem.java` um das Rucksackproblem zu lösen. Verwenden Sie keine Schleifen, nur rekursive Aufrufe.

3. Implementieren Sie eine iterative Variante des Algorithmus in `solveIterative`, die ohne rekursive Aufrufe auskommt und stattdessen eine Schleife und eine Stack-Datenstruktur verwendet.

4. Testen Sie Ihre Implementierungen mit den vorgegebenen Testfällen. *Hinweis:* Öffnen Sie `RucksackProblemTest.java` im Editor. Wählen Sie im Menü: *Run, Run As, JUnit Test*. Alle Tests die mit *rucksack* beginnen, testen die Klasse `Rucksack`, die anderen die Klasse `RucksackProblem`.

5. Beantworten Sie folgende Fragen im Kopf von `RucksackProblem.java` an den mit **TODO:** markierten Stellen:

   a) Was sind die Laufzeitkomplexitäten T(n) der rekursiven und der iterativen Version des Algorithmus, wobei n die Anzahl der packbaren Objekte ist? Geben Sie die Komplexität in O-Notation an.

   b) Verwenden Sie TPTP um `RucksackProblem.main()` zu profilen. Probieren Sie verschiedene Werte für die Variable numObjects in main() aus und messen Sie die Laufzeit von `solveRecursive(int)` und `solveIterative` sowie die Anzahl der rekursiven Aufrufe von `solveRecursive(int)`. Bis zu welcher Größe können Sie auf Ihrem Rechner gehen? Tragen Sie mindestens 5 verschiedene, aussagekräftige Resultate in der Tabelle ein.

**Abgabe.** Reichen Sie Ihre beiden fertig bearbeiteten Dateien `Rucksack.java` und `RucksackProblem.java` bis zum Abgabetermin in moodle ein. Die Abgabe ist in Gruppen von bis zu drei Personen möglich. **Ein** Mitglied der Gruppe reicht die Lösung zusammen mit einem Kommentar der folgenden Form ein: *Die Übung wurde bearbeitet von der Gruppe X, Y, Z; die Abgabe erfolgte durch X*. **Jedes andere** Mitglied der Gruppe reicht *anstelle* der Lösung nur diesen (identischen) Kommentar ein.

Bitte stellen Sie sicher, dass Sie und ggf. Ihre Gruppenmitglieder diese Regeln befolgen; wird weder eine Lösung noch der obenstehende Kommentar eingereicht, so vergibt der Tutor auch keine Punkte!

***Hinweis:*** *Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Mit der Abgabe einer Programmieraufgabe bestätigen Sie, dass Sie bzw. Ihre Lerngruppe die alleinigen Autoren sind.*