FAKULTÄT FÜR INFORMATIK
Faculty of Informatics

Objektorientierte Programmiertechniken
LVA 185.A01, VL 2.0, 2011 W

6. Übungsaufgabe

Themen:
dynamische Typinformation, homogene Übersetzung von Generizität

Termine:

Ausgabe:	30.11.2011
reguläre Abgabe:	07.12.2011, 13:45 Uhr
nachträgliche Abgabe:	14.12.2011, 13:45 Uhr

Abgabeverzeichnis:
Aufgabe6

Programmaufruf:
java Test

Grundlage:
Skriptum bis Seite 124, Schwerpunkt auf den Abschnitten 3.3.1 und 3.3.2

Aufgabe

Welche Aufgabe zu lösen ist:
In automatisierten Fabriken erfolgt die Produktion unter Verwendung von Robotern. Jede Fabrik hat einen eindeutigen, unveränderlichen Namen und verwaltet eine Menge von Robotern. Jeder Roboter hat eine eindeutige, unveränderliche Nummer und ist entweder ein fixer Schwenkarmroboter oder ein mobiler Raupenroboter. In jedem Roboter wird gespeichert, seit wievielen Stunden der Roboter bereits in Betrieb ist. Bei einem fixen Schwenkarmroboter wird zusätzlich abgespeichert, wieviele Rotationen der Schwenkarm ausgeführt hat (ganze Zahl), und bei einem mobilen Raupenroboter wird abgespeichert, wieviele Meter Wegstrecke er zurückgelegt hat (Gleitkommazahl). Jeder Roboter kann für seinen Einsatzzweck umgerüstet werden und entweder als Schweißroboter oder als Lackierroboter eingesetzt werden. Von jedem Schweißroboter ist die Arbeitstemperatur des Schweißgeräts in Grad (ganze Zahl) bekannt, von jedem Lackierroboter die Fassungskapazität seines Lackbehälters in Liter (Gleitkommazahl). Zu jedem Zeitpunkt wird ein Roboter höchstens für eine Art von Aufgabe eingesetzt.

Entwickeln Sie Java-Klassen bzw. Interfaces zur Darstellung von Robotern auch für unterschiedliche Einsatzarten. Folgende

Funktionalität soll unterstützt werden:

- Erzeugen eines Roboters.
- Erhöhen der Betriebsstunden.
- Auslesen der Betriebsstunden.
- Erhöhen und Auslesen der Anzahl der ausgefürten Rotationen eines Schwenkarmroboters.
- Erhöhen und Auslesen der zurückgelegten Wegstrecke eines Raupenroboters.
- Ändern der Einsatzart eines Roboters, wobei Informationen über frühere Einsatzarten dieses Roboters verloren gehen.

Schreiben Sie eine Klasse *Fabrik*, die Informationen über eine Fabrik verwaltet und statistische Auswertungen über diese Fabrik ermöglicht. Jede Fabrik wird über den unveränderlichen Namen identifiziert. Folgende Methoden sollen unterstützt werden:

- Erzeugen einer Fabrik.
- Einfügen von Robotern in eine Fabrik.
- Entfernen von Robotern aus einer Fabrik.
- Ändern der Informationen über Roboter wie oben beschrieben.
- Methoden zum Berechnen folgender statistischer Werte:
    - Die durchschnittliche Anzahl der Betriebstunden aller Roboter einer Fabrik – alle Roboter zusammen und zusätzlich aufgeschlüsselt nach den Einsatzarten (Schweißroboter oder Lackierroboter).
    - Die durchschnittliche Anzahl der Betriebstunden aller Roboter einer Fabrik aufgeschlüsselt nach den Robotertyp (Schwenkarmroboter oder Raupenroboter).
    - Die durchschnittliche Anzahl der Rotationen aller Schwenkarmroboter einer Fabrik – alle zusammen und zusätzlich aufgeschlüsselt nach den Einsatzarten (Schweißroboter oder Lackierroboter).
    - Die durchschnittliche zurückgelegte Wegstrecke aller Raupenroboter einer Fabrik – alle zusammen und zusätzlich aufgeschlüsselt nach den Einsatzarten.
    - Die minimale und maximale Arbeitstemperatur aller Schweißroboter insgesamt und aufgeschlüsselt nach Robotertyp (Schwenkarmroboter oder Raupenroboter).
    - Die durchschnittlich zurückgelegte Wegstrecke aller Raupenroboter insgesamt und aufgeschlüsselt nach Robotertyp.

Die Klasse *Test* soll die wichtigsten Normal- und Grenzfälle (nicht interaktiv) überprüfen und die Ergebnisse in allgemein verständlicher Form in der Standardausgabe darstellen. Machen Sie unter anderem Folgendes:

- Erstellen Sie eine Menge von Fabriken von jeweils einigen Robotern – wirklich eine *Menge* von Fabriken (eine Form von Collection), nicht nur eine Ansammlung einzelner Variablen. Jede Fabrik in der Menge soll über ihren eindeutigen Namen

angesprochen werden können, und jeder Roboter einer Fabrik über seine eindeutige Nummer.
- Fügen Sie zu einigen Fabriken einzelne Roboter hinzu, entfernen Sie einzelne Roboter, und ändern Sie die Informationen zu einzelnen Robotern, wobei Sie Roboter und Fabriken nur über deren Nummern und Namen ansprechen.
- Berechnen Sie die statistischen Werte aller Fabriken (wie oben beschrieben).

Generizität, Arrays und vorgefertigte Container-Klassen dürfen zur Lösung dieser Aufgabe nicht verwendet werden. Vermeiden Sie mehrfach vorkommenden Code für gleiche oder ähnliche Programmteile.

Warum die Aufgabe diese Form hat:
Diese Aufgabenstellung lässt Ihnen mehr Entscheidungsspielraum als die drei vorangegangenen. Die gleichzeitige Unterscheidung von Schwenkarmrobotern und Raupenrobotern sowie zwischen unterschiedlichen Einsatzarten stellt beispielsweise eine Schwierigkeit dar, für die es mehrere sinnvolle Lösungsansätze gibt. Sie werden irgendeine Form von Container selbst erstellen müssen, wobei die genaue Form und Funktionalität nicht vorgegeben ist. Da Container an mehreren Stellen benötigt werden, wäre die Verwendung von Generizität sinnvoll. Dadurch, dass Sie Generizität nicht verwenden dürfen und trotzdem mehrfache Vorkommen ähnlichen Codes vermeiden sollen, werden Sie gezwungen, Techniken ähnlich denen einzusetzen, die der Compiler zur homogenen Übersetzung von Generizität verwendet. Vermutlich sind Typumwandlungen kaum vermeidbar. Sie sollen dadurch ein tieferes Verständnis des Zusammenhangs zwischen Generizität und Typumwandlungen bekommen.

Was im Hinblick auf die Beurteilung zu beachten ist:
Der Schwerpunkt bei der Beurteilung liegt auf der vernünftigen Verwendung von dynamischer und statischer Typinformation. Kräftige Punkteabzüge gibt es für

- die Verwendung von Generizität bzw. von Arrays oder vorgefertigten Container-Klassen
- mehrfach vorkommende gleiche oder ähnliche Programmteile (wenn vermeidbar)
- den unnötigen Verlust an statischer Typsicherheit
- Verletzungen des Ersetzbarkeitsprinzips bei Verwendung von Vererbungsbeziehungen (also Vererbungsbeziehungen, die keine Untertypbeziehungen sind)
- und mangelhafte Funktionalität des Programms.

Punkteabzüge gibt es unter anderem auch für mangelhafte Zusicherungen und falsche Sichtbarkeit.

Wie die Aufgabe zu lösen ist:
Es wird empfohlen, die Aufgabe zuerst mit Hilfe von Generizität zu

lösen und in einem weiteren Schritt eine homogene Übersetzung der Generizität (wie im Skriptum beschrieben) händisch durchzuführen. Durch diese Vorgehensweise erreichen Sie eine statische Überprüfung der Korrektheit vieler Typumwandlungen und vermeiden den unnötigen Verlust an statischer Typsicherheit.

Zur Lösung dieser Aufgabe ist die Verwendung von Typumwandlungen ausdrücklich erlaubt. Versuchen Sie trotzdem, die Anzahl der Typumwandlungen klein zu halten und so viel Typinformation wie möglich statisch vorzugeben. Das hilft Ihnen dabei, die Lösung überschaubar zu halten und einen unnötigen Verlust an statischer Typsicherheit zu vermeiden. Gehen Sie auch möglichst sparsam mit dynamischen Typabfragen und Ausnahmebehandlungen um.

Achten Sie darauf, dass Sie Divisionen durch 0 vermeiden. Führen Sie zumindest einen Testfall ein, bei dem eine statistische Auswertung ohne entsprechende Vorkehrungen eine Exception aufgrund einer Division durch 0 auslösen würde.

Bedenken Sie, dass es mehrere sinnvolle Lösungsansätze für diese Aufgabe gibt. Wenn Sie einmal einen gangbaren Weg gefunden haben, bleiben Sie dabei, und vermeiden Sie es, zu viele Möglichkeiten auszuprobieren. Das könnte Ihnen viel Zeit kosten, ohne die Lösung zu verbessern.

Die Art der Verwendung eines Roboters für unterschiedene Einsatzzwecke kann sich im Laufe der Zeit ändern. Am besten stellt man solche Beziehungen über *Rollen* dar: Für jede Art von Roboter gibt es eine eigene Klasse mit den für die jeweilige Art typischen Daten, und ein gemeinsamer Obertyp ermöglicht den Zugriff auf diese Daten auf einheitliche Weise. In jedem Roboter gibt es einen Zeiger auf die aktuelle Einsatzart (= die Rolle, die der Roboter gerade spielt). Wenn sich die Art ändert, braucht nur dieser Zeiger neu gesetzt zu werden. Durch geschickte Auswahl der Methoden einer Rolle sind die meisten Fallunterscheidungen vermeidbar, das heißt, Fallunterscheidungen werden durch dynamisches Binden ersetzt.

Was im Hinblick auf die Abgabe zu beachten ist:
Verzichten Sie auch bei der Lösung dieser Aufgabe auf die Verwendung von packages und Verzeichnissen innerhalb des Abgabeverzeichnisses *Gruppe/Aufgabe6*. Schreiben Sie (abgesehen von geschachtelten Klassen) nicht mehr als eine Klasse in jede Datei.

Anfang | HTML 4.01 | letzte Änderung: 2011-11-30 (Puntigam)