TU Berlin
Software and Business Engineering
Prof. Dr. Ingo Weber
Dr. Tobias Heindel
Richard Hobeck
Martin Grambow

WiSe 2020/21
Aufgabe 6
Block 2
14. Dezember 2020

Abgabetermin: 3. Januar 2021

Aufgabe 6 *Einführung in die Programmierung mit JAVA*

Hinweise

- Die Abgabe dieser Übungsaufgaben muss bis spätestens Sonntag, den 3. Januar 2021 um 23:59 Uhr im ISIS-Kurs erfolgt sein. Es gelten die Ihnen bekannten Übungsbedingungen.
- Lösungen zu diesen Aufgaben sind als gezippter Projektordner abzugeben. Eine Anleitung zum Zippen von Projekten finden Sie auf der Seite des ISIS-Kurses. *Bitte benutzen Sie einen Dateinamen der Form* `VornameNachname.zip`.
- Bitte beachten Sie, dass Abgaben im Rahmen der Übungsleistung für die Zulassung zur Klausur relevant sind. Durch Plagiieren verwirken Sie sich die Möglichkeit zur Zulassung zur Klausur in diesem Semester.

In dieser Aufgabe werden Sie ein kleines Software-Projekt zum Thema Paketzustellung bearbeiten. Weil aufgrund der Weihnachtszeit mit viel mehr Lieferungen und Bestellungen zu rechnen ist, stellt der Weihnachtsmann selbst die Lieferungen zu und deshalb müssen Straßen nicht berücksichtigt werden.

Aufgabe 6.1 *Briefe, Päckchen und Lieferungen: code-review* (1 Punkt)

Machen Sie sich mit den Klassen `Delivery`, `Letter`, und `Parcel` im Paket

```
package de.tuberlin.mcc.prog1.logistics.deliveries
```

der Vorgabe vertraut, die auf der Seite des ISIS-Kurses bereitgestellt ist.

Aufgabe 6.2 *`DeliveryManager` & `Delivery`* (2 Punkte)

Für den Zustellalgorithmus müssen die Klassen `DeliveryManager` und `Delivery` zunächst erweitert werden. Implementieren Sie als erstes die Methode `removeDelivery()` in der Klasse `DeliveryManager`. Die Methode soll eine gegebene Lieferung im Array der registrierten Lieferungen suchen und diese aus dem Array entfernen. Wird die gegebene Lieferung nicht gefunden, soll der Wert `null` zurückgegeben werden. Wird die Lieferung gefunden, so soll das entsprechende Element im Array auf `null` gesetzt werden und die entfernte Lieferung soll zurückgegeben werden. Für das Suchen und Vergleichen von Lieferungen müssen Sie, je nach Lösungsansatz, die Methoden `equals()` und/oder `hashCode()` in der Klasse Delivery überschreiben.
Die Methode `registerDeliveries()` fügt neue Lieferungen am Ende des Arrays `deliveries` an und vergrößert dieses Array dadurch immer weiter. Andererseits entfernt die Methode `removeDelivery()` Lieferungen aus dem Array und setzt die entsprechenden Werte auf `null`. Im späteren Programmablauf würde dies dazu führen, dass am Anfang des Arrays `null`-Werte stehen und erst gegen Ende des Arrays Referenzen auf Lieferungen vorhanden sind. Verhindern Sie dies, indem Sie die Methode `registerDeliveries()` so anpassen, dass das Array auch schrumpft, wenn es `null`-Werte am Anfang des Arrays gibt; `null`-Werte, die in der Mitte des Arrays auftreten, müssen Sie nicht berücksichtigen.
Implementieren Sie außerdem die Methode `countDeliveriesForLocation()`, die die Anzahl von Lieferungen, die an einem gegebenen Absendeort vorliegen, ermitteln soll.

Aufgabe 6.3 *Zustellungsalgorithmus* **(2 Punkte)**

Implementieren Sie folgenden einfachen Zustellungsalgorithmus in der Klasse `Santa`, indem Sie die Methode `computeNextOperation()` anpassen, und zwar im Paket

`de.tuberlin.mcc.prog1.logistics.santa`.

Ihren aktuellen Algorithmus können Sie testen, bzw. simulieren, indem Sie die ausführbare Klasse `LogisticsManager` starten. Die Methode erhält folgende Parameter:

- `x`: die aktuelle x-Position des Weihnachtsmanns
- `y`: die aktuelle y-Position des Weihnachtsmanns
- `registeredDeliveries`: ein Array mit verfügbaren Lieferungen
- `inventory`: ein Array mit Lieferungen, die der Weihnachtsmann mit sich führt

Die Methode soll, je nach Situation, ein Kommando als String zurückgeben. Folgende Kommandos sind zulässig:

- `N`, `S`, `W`, `E`: für Bewegung des Weihnachtsmanns auf der Karte um eine Position in die entsprechende Himmelsrichtung
- `G` *`idx`*: für das Abholen der Lieferung mit Index *idx* und Hinzufügen zum Inventar des Weihnachtsmanns[1]
- `D` *`idx`*: für die Lieferung mit Index *idx* an den entsprechenden Empfänger[2]

Implementieren Sie folgenden einfachen Zustellalgorithmus:

1. Wenn eine Lieferung im Inventar ist, dann stelle sie zu, wenn die aktuelle Position mit der Position des Emfängers der Lieferung identisch ist.
2. Wenn eine Lieferung im Inventar ist und die aktuelle Position nicht mit der Position des Empfängers identisch ist, dann gehe einen Schritt dichter an diese Position.
3. Wenn keine Lieferung im Inventar ist, dann hole die erste Lieferung, wenn die aktuelle Position mit der Position des Senders der ersten Lieferung identisch ist.
4. Wenn keine Lieferung im Inventar ist und die aktuelle Position nicht mit der Position des Senders der ersten Lieferung identisch ist, dann gehe einen Schritt dichter an diese Position.

Sie können die Klasse `Santa` beliebig erweiteren und weitere Klassen-/Instanzvariablen und Klassen-/Instanzmethoden hinzufügen, um diesen Algorithmus zu implementieren.

Aufgabe 6.4 *Simulation* **(1 Punkt)**

Testen Sie Ihren Algorithmus durch Ausführung der Klasse `LogisticsManager`. Passen Sie Ihre Simulation individuell an, indem Sie weitere Orte auf der Karte hinzufügen oder bestehende anpassen. Außerdem können Sie in der `main()`-Methode die Anzahl der Weihnachtsmänner erhöhen. Funktioniert Ihr Algorithmus auch mit mehreren Weihnachtsmännern?

Aufgabe 6.5 *Super-Solo-Santa* **(2 Zusatzpunkte)**

Optimieren Sie den Algorithmus für den Fall von einem Weihnachtsmann.

Aufgabe 6.6 *Dream Team „Santa“* **(2 Zusatzpunkte)**

Optimieren Sie den Algorithmus für den Fall von mehreren Weihnachtsmännern.

[1] Der Weihnachtsmann muss dafür auf der Position des Senders der entsprechenden Lieferung stehen.
[2] Der Weihnachtsmann muss dafür auf der Position des Empfängers der entsprechenden Lieferung stehen.