HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN · FH MÜNCHEN

Algorithmen und Datenstrukturen (IB) Praktikum
Aufgabenblatt 3 – Disjoint Sets

Ausgabe: 4. April 2018
Abgabe: 10. April 2018, 18:00 Uhr

1 Disjoint Sets (Pflichtaufgabe)

Sie haben in der Vorlesung den abstrakten Datentyp *Disjoint Set* kennengelernt sowie eine Vereinfachung für Ganzzahlen in einem festen Wertebereich von 0 bis size − 1.

1.1 Auswertung von Disjoint Set

Betrachten Sie die folgenden Verwendungen dieser Vereinfachung. **Geben Sie bei `union`-Aufrufen den neuen Zustand der Struktur, bei `find`-Aufrufen das Ergebnis an!** Geben Sie Ihre Lösung als Foto/Scan oder als elektronisches Dokument in Moodle ab.

Operation	*Ergebnis*
Erstellung(8)	{0}, {1}, {2}, {3}, {4}, {5}, {6}, {7}
`find(2)`	
`union(2,3)`	
`find(3)`	
`union(4,5)`	
`union(3,7)`	
`union(0,6)`	
`find(0)`	
`union(0,7)`	
`find(2)`	

Operation	*Ergebnis*
Erstellung(8)	{0}, {1}, {2}, {3}, {4}, {5}, {6}, {7}
`union(4, 2)`	
`union(1,3)`	
`union(5,7`	
`union(0,6)`	
`union(7,3)`	
`find(5)`	
`find(2)`	
`union(0,7)`	
`find(6)`	

1.2 Implementierung Integer Disjoint Sets

In Moodle ist ein Maven-Projekt bereitgestellt, indem Sie im Paket `hm.edu.cs.algdat.disjointset` ein Interface `DisjointIntegerSets` finden, das so einem abstrakten Datentyp entspricht. Daneben finden Sie mit der Klasse `MyDisjointIntegerSets` eine unvollständige Implementierung und eine Testklasse `MyDisjointIntegerSetsTest`.

Ergänzen Sie die Implementierung der Klasse `MyDisjointIntegerSets` entsprechend der Beschreibung aus der Vorlesung. Die Tests sollten dann alle erfüllt sein. Tipp: Es gibt sehr viele Möglichkeiten, diesen Typ zu implementieren. Am einfachsten ist es, ein Array mit aktuellen Repräsentanten für die Zahlen zu erstellen und bei `union`-Operationen zu aktualisieren. Es ist nicht gefordert, eine besonders effiziente/performante Implementierung zu liefern! Es reicht, wenn Sie die Klasse `MyDisjointIntegerSets` in Moodle hochladen.

2 Verwendung von Disjoint Sets zur Labyrinthgenerierung (Optional)

In der Vorlesung oder im Praktikum haben Sie gesehen, dass klassische Labyrinthe (ohne Wege, die im Kreis führen) erstellt werden können, indem man auf einem Gitter sukzessive zufällig Wände entfernt, die zwei Felder trennen *zwischen noch keine Verbindung besteht*.

In dem Maven-Projekt befindet sich eine Beispielanwendung `MazeApplication` im Paket `hm.edu.cs.algdat.maze.vis`, die die schrittweise Erstellung eines Labyrinths visualisiert. An dieser Klasse müssen Sie nichts ändern. Die Durchführung der Schritte (Erstellen aller Wände und sukzessives Entfernen) ist in der Klasse `MazeCreator` im Paket `hm.edu.cs.algdat.maze` implementiert. Es fehlt allerdings eine sinnvolle Überprüfung, ob zwei Felder bereits verbunden sind, so dass alle Wände nach und nach entfernt werden (siehe Abbildung 1).

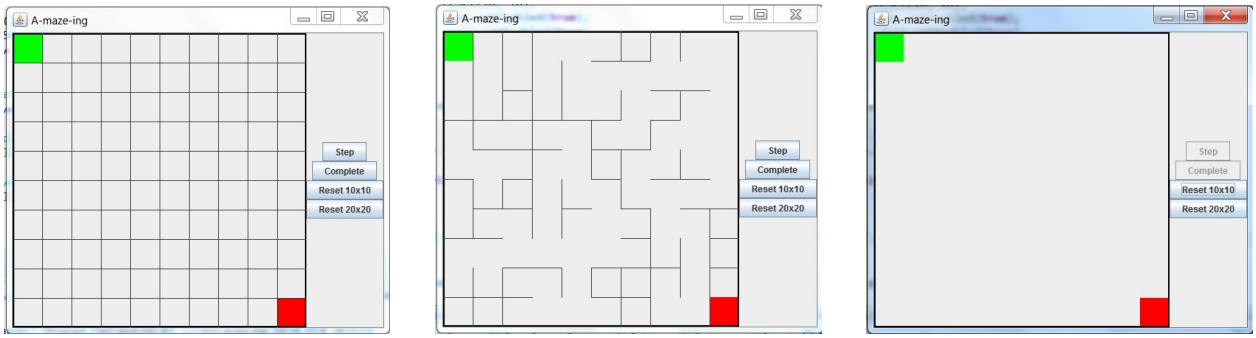

Abbildung 1: Ohne sinnvolle Überprüfung werden alle Wände nach und nach entfernt.

Ergänzen Sie die Klasse MazeCreator so, dass nur Wände entfernt werden, die bis dahin noch nicht verbunden sind. Dazu müssen Sie im Konstruktor eine geeignete Instanz Ihrer Implementierung von `MyDisjointIntegerSets` erstellen und in der Methode `step()` geeignet verwenden, so dass ein korrektes Labyrinth erstellt wird (siehe Abbildung 2).

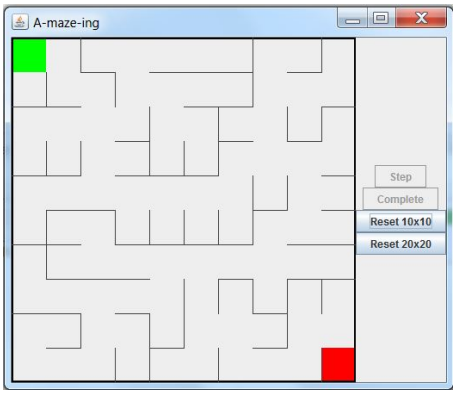

Abbildung 2: Durch sinnvollen Einsatz einer Disjoint-Sets-Struktur kann ein Labyrinth erzeugt werden!

Tipp: Verwenden Sie eine Instanz von `MyDisjointIntegerSets` mit einer der Anzahl an Elementen, die der Anzahl der Felder des Labyrinths entspricht und bilden Sie Felder (Zeile und Spalte) auf ganze Zahlen ab, indem Sie eine beliebige Nummerierung der Felder zugrundelegen. Machen Sie sich klar, inwiefern *verbundene Felder* sich durch eine Partitionierung darstellen lässt. Wenn Sie Feedback zu Ihrer Lösung wünschen, geben Sie die Klasse MazeCreator in Moodle ab!