Protokoll mit Beschreibung der Datenstruktur:

-Datenstruktur:

Unsere Tabelle ist ein Array von Listen, die jeweils Paare aus einer Zahl und einem Aktienobjekt enthalten. Das Array hat eine feste Größe von 2609(1,3-mal die Anzahl an Elementen und es ist Primzahl) Elementen, die als „hashGroups“ bezeichnet werden. Jedes Element des Arrays ist eine Liste, die mehrere Paare speichern kann.

Die Zahl in jedem Paar ist der Schlüssel, der aus dem Namen oder dem Kürzel der Aktie erzeugt wird. Das Aktienobjekt in jedem Paar ist der Wert, der Informationen über die Aktie enthält.

Diese Datenstruktur ermöglicht es, ein Paar anhand seines Schlüssels schnell zu finden oder einzufügen.

-Kollisionserkennung:

Eine Kollision tritt auf, wenn die Hashfunktion denselben Index für mehrere Schlüssel erzeugt. Wir machen eine Verkettung von Kollisionen. Das bedeutet, dass jede Zelle der Hashtabelle eine Liste von Paaren aus Schlüssel und Wert enthält. Wenn ein neuer Wert mit demselben Index eingefügt wird, wird er an das Ende der Liste angehängt.

-Hashfunktion:

Unser „createKey“-Funktion verwendet den Algorithmus von „hashCode“ aus Java, um einen Hashwert für einen String zu berechnen. Sie multipliziert den bisherigen Hashwert mit 31 und addiert den ASCII-Wert des aktuellen Zeichens. Sie begrenzt den Hashwert auf 48 Bits, indem sie ihn mit 0xFFFFFFFFFFFF verrundet.

Dann in der „hashFunction“-Funktion verwenden wir den Modulo-Operator, um einen Index für die Hashtabelle zu berechnen. In diesem Fall haben wir 43 genommen, da es eine Primzahl ist. Dieser Index bestimmt, in welcher Zelle der Hashtabelle der Wert gespeichert wird.

-Aufwandschätzung:

Suchen: In einem Array müssten wir jedes Element der Array einlesen und vergleichen. Anstatt werden nun nur die Werte, die dieselbe Key haben überprüft. Mit einer guten Hashfunktion die Kollisionen minimiert wird in der Praxis viel schneller als ein Array.

Einfügen: Einfügen ist bei Arrays sehr viel langsamer auch. Man muss bis am Ende kommen und etwas hinzufügen zu können. Bei der Hashtabelle passiert es nur im Fall von Kollisionen, sonnst wird es nur O(1) sein.

Löschen: In einem Array ist das Löschen viel schwieriger, da man alle Werte dahinten auch nach vorne verschieben muss. Mit einer Liste wie in unserer Lösung kann man viel einfacher Knoten verbinden und die Laufzeit für die Operation effektiv reduzieren.