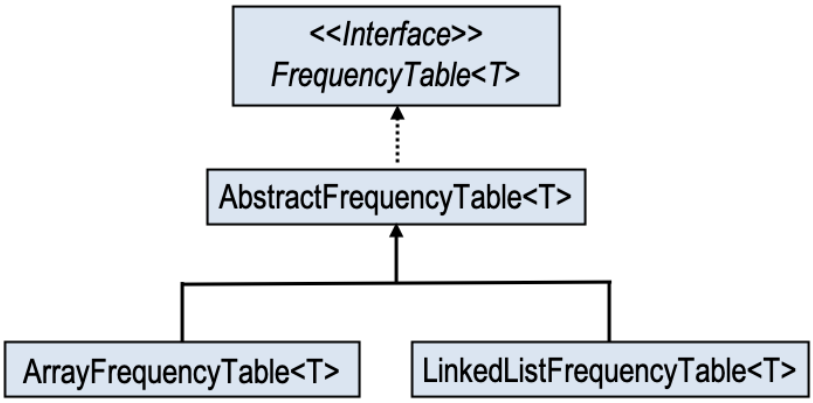
Hochschule Konstanz
Fakultät Informatik

Aufgabenblatt 4

Teil 1

Ändern Sie die in der Aufgabe1 und 2 realisierten Typen in generische Typen um. Anstatt die Häufigkeiten von Strings zu verwalten, soll es jetzt möglich sein, die Häufigkeiten von Elementen eines beliebigen Typs zu verwalten. Die Klasse `Word` aus Aufgabe 1 und 2 zur Speicherung von Wort-Häufigkeits-Paaren soll nun sinnvollerweise in eine generische Klasse `Element<T>` (Element-Häufigkeits-Paar) geändert werden. Testen Sie Ihre Klasse, indem Sie die beiden bisherigen Testklassen entsprechend anpassen.



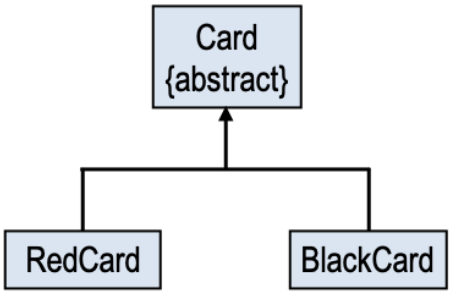
Teil 2

Das Interface `FrequencyTable<T>` soll möglichst flexibel sein. Dazu soll geprüft werden, ob das PECS-Prinzip eingehalten worden ist. Beachten Sie dabei auch das in der Vorlesung besprochene Interface `Collection<E>.`

Implementieren Sie dazu die Klassen `Card`, `RedCard` und `BlackCard` für Spielkarten. Eine Spielkarte hat eine Farbe (Karo, Herz, Pique und Kreuz) und einen Wert (sieben, acht, neun, zehn, Bube, Dame König, Ass). Die Klassen bieten einen Konstruktor und entsprechende getter-Methoden an. Bei `RedCard` darf die Kartenfarbe nur rot und bei `BlackCard` nur schwarz sein.

Hinweis: Definieren Sie die Klasse `Card` als abstrakt. Es können dann nur `RedCard`- und `BlackCard`-Instanzen erzeugt werden.

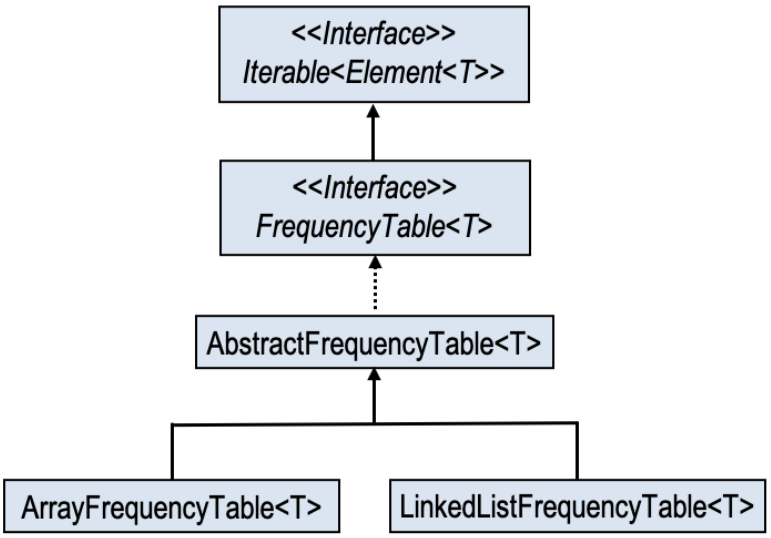
Implementieren Sie auch die Methoden `toString` und `equals`.



Testen Sie Ihre Klasse mit dem Testprogramm auf der Web-Seite.

Teil 3

Das Interface `FrequencyTable<T>` soll nun das Interface `Iterable<Element<T>>` erweitern.



Erweitern Sie `ArrayFrequencyTable<T>` und `LinkedListFrequencyTable<T>` jeweils um ein Iteratorkonzept. Die remove()-Methode muss nicht unterstützt werden.

Achten Sie nun darauf, dass alle Methoden aus `AbstractFrequencyTable<T>` mit einer foreach-Schleife implementiert sind (d.h. auf Basis von Iteratoren).