Fachhochschule
Nordwestschweiz

01 Java Collections

Algorithmen und Datenstrukturen 2

Fachhochschule
Nordwestschweiz

Set

- Ein Element ist entweder drin oder nicht
- Mögliche Operationen:
  Hinzufügen, Entfernen, Suchen
  (Vereinigen, Schnittmenge, etc.)
- Andere Bezeichnungen:
  Menge (math.)

Bag

- Ein Element kann mehrfach enthalten sein
- Mögliche Operationen:
  Hinzufügen, Entfernen, Suchen
- Andere Bezeichnungen:
  Multimenge, Multiset

Set vs Bag

Ein Element wird dreimal hinzugefügt:

- Wie oft ist es im Set enthalten?

- Wie oft ist es im Bag enthalten?

Nun wird es einmal gelöscht:

- Wie oft ist es im Set enthalten?

- Wie oft ist es im Bag enthalten?

Set vs Bag

Ein Element wird dreimal hinzugefügt:

- Wie oft ist es im Set enthalten?
  1x. Das Element wurde beim ersten Mal eingefügt. Danach nicht mehr.
- Wie oft ist es im Bag enthalten?
  3x.

Nun wird es einmal gelöscht:

- Wie oft ist es im Set enthalten?
  Das Element ist nicht mehr enthalten.
- Wie oft ist es im Bag enthalten?
  Noch 2x.

Collections

Collection = Sammlung von Elementen derselben Klasse (meistens)

- Eine Collection kann Duplikate erlauben oder ausschliessen.
- Eine Collection kann sortiert oder unsortiert sein.

		Sortierung	
		Sortiert	**Unsortiert**
Semantik	**Bag-Semantik**	SortedBag	UnsortedBag
	Set-Semantik	SortedSet	UnsortedSet

Fachhochschule
Nordwestschweiz

Arbeitsblatt

Lösen Sie die Aufgabe 1.

Lernziele heute

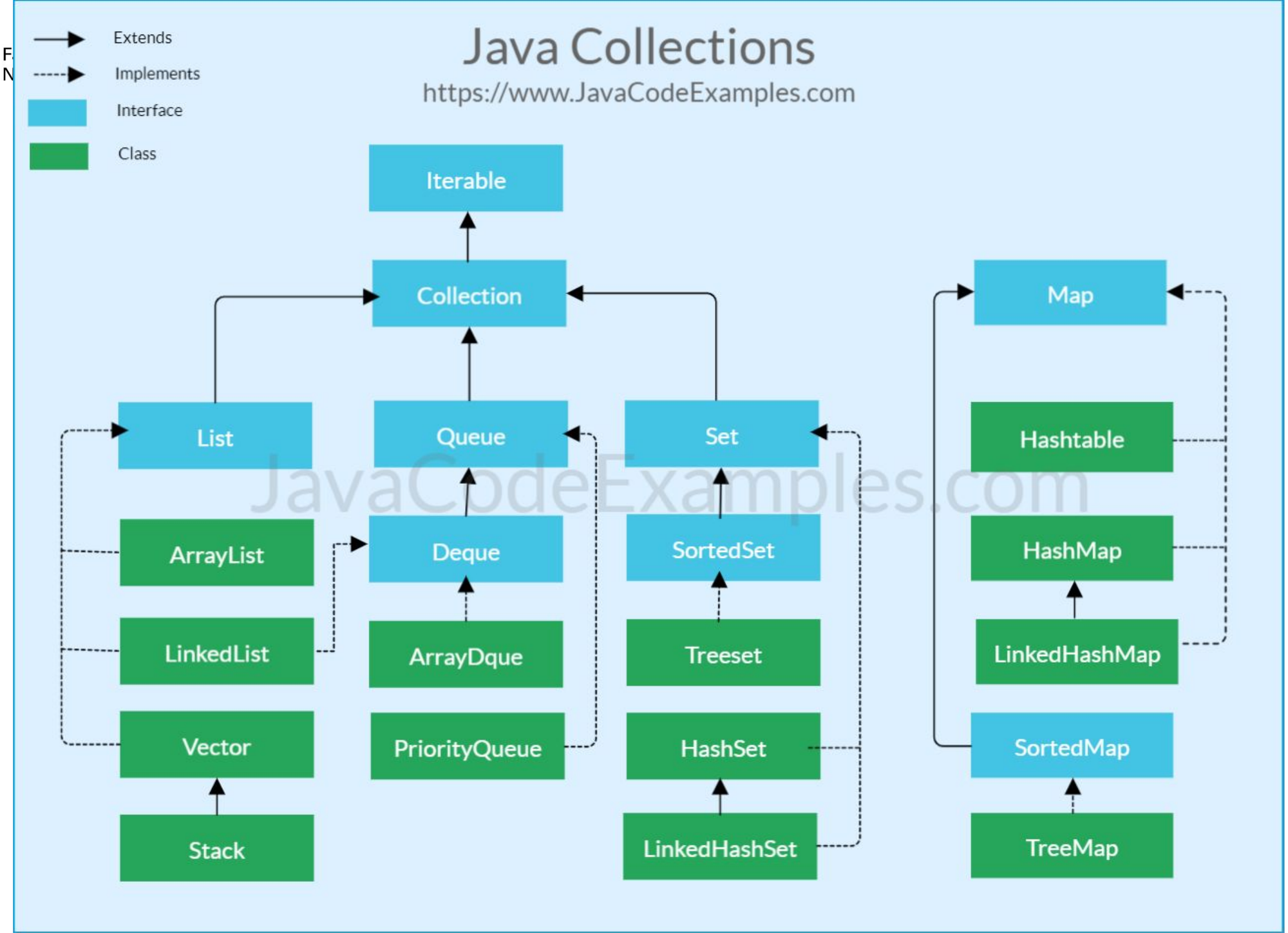
- Sie kennen den Unterschied zwischen *Bag-Semantik* und *Set-Semantik*.
- Sie kennen Vor- und Nachteile der sortierten und unsortierten Datenablage im Array.
- Sie wählen für gegebene Anforderungen eine geeignete Kombination.
- Sie implementieren einfache Array-basierte *Collections* (Sammlung von Elementen) für Bag-Semantik und Set-Semantik.
- Sie ermitteln für eine gegebene Implementierung die Komplexität der wichtigsten Zugriffsoperationen.

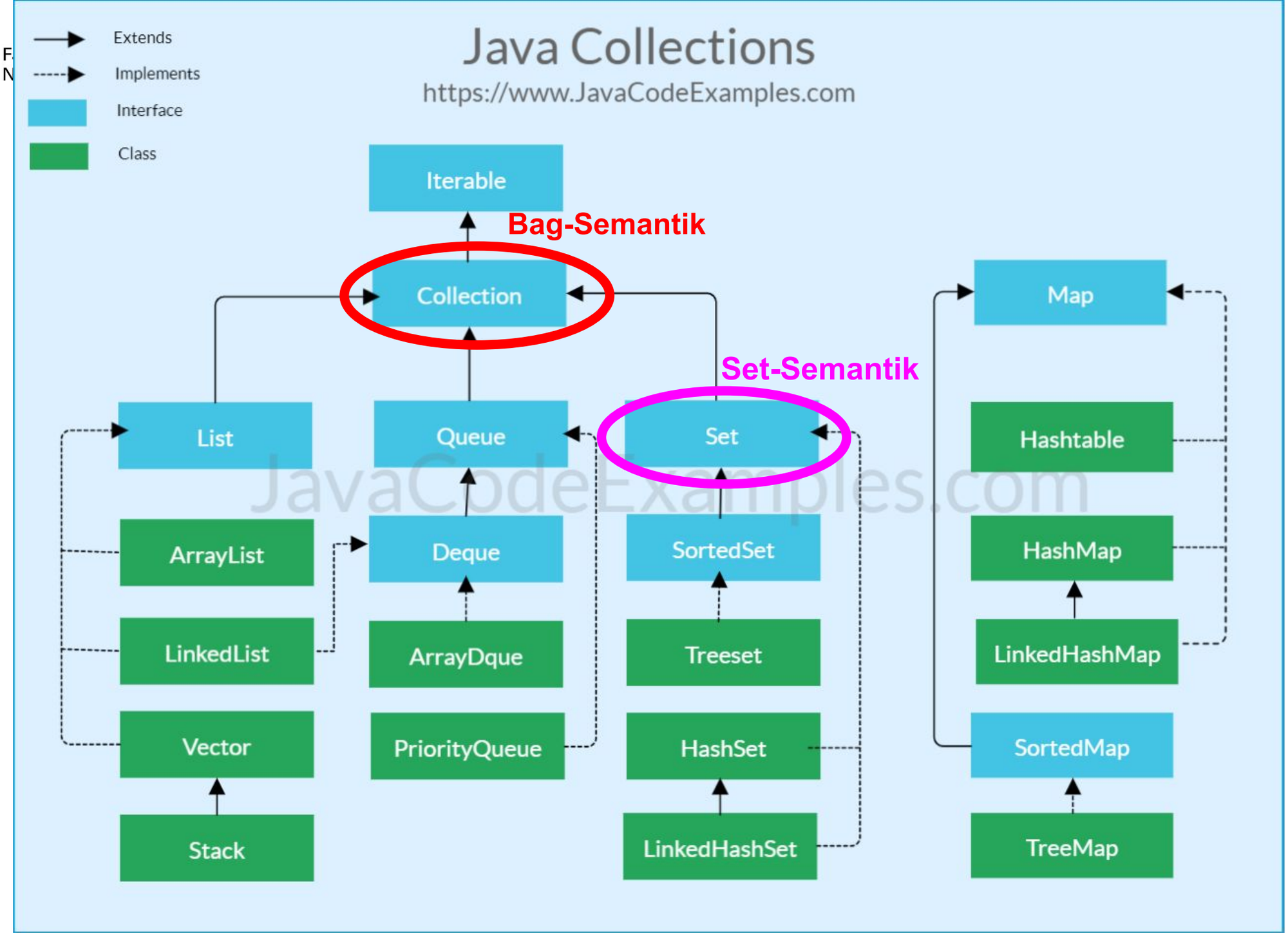
Java Collection Framework

Zugriffsoperationen auf eine Collection:

- Hinzufügen (*add*)
- Entfernen (*remove*)
- Suchen (*contains*)

https://docs.oracle.com/javase/11/docs/api/java/util/Collection.html


Java Collections
https://www.JavaCodeExamples.com
Extends
Implements
Interface
Class
Iterable
Collection
Map
List
Queue
Set
Hashtable
ArrayList
Deque
SortedSet
HashMap
LinkedList
ArrayDque
Treeset
LinkedHashMap
Vector
PriorityQueue
HashSet
SortedMap
Stack
LinkedHashSet
TreeMap
JavaCodeExamples.com


Java Collections
https://www.JavaCodeExamples.com
Extends
Implements
Interface
Class
Iterable
Bag-Semantik
Collection
Set-Semantik
Map
List
Queue
Set
Hashtable
ArrayList
Deque
SortedSet
HashMap
LinkedList
ArrayDque
Treeset
LinkedHashMap
Vector
PriorityQueue
HashSet
SortedMap
Stack
LinkedHashSet
TreeMap
JavaCodeExamples.com

Standardoperationen im Collection und Set Interface

`boolean add(E e), boolean remove(E e)`

Keine Exception:

- Der Aufruf war gemäss Spezifikation erfolgreich.
- Sagt *nichts* aus, ob die Collection verändert wurde.

Boolean-Rückgabewert:

- *True*: Collection wurde verändert.
- *False*: Collection wurde nicht verändert.

Standardoperationen im Collection und Set Interface

`boolean add(E e), boolean remove(E e)`

Keine Exception:

- Der Aufruf war gemäss Spezifikation erfolgreich.
- Sagt *nichts* aus, ob die Collection verändert wurde.

Boolean-Rückgabewert:

- *True*: Collection wurde verändert.
- *False*: Collection wurde nicht verändert.

`boolean contains(Object o)`

Keine Exception:

- Der Aufruf war gemäss Spezifikation erfolgreich.
- Sagt nichts aus, ob das Objekt in der Collection ist.

Boolean-Rückgabewert*:*

- *True*: Objekt ist in Collection.
- *False*: Objekt ist nicht in Collection.

Fachhochschule
Nordwestschweiz

Arbeitsblatt

Lösen Sie die Aufgabe 2.

Aufgabe 2 – Lösungen

1. c.add(1) gibt false zurück. Mögliche Erklärungen:
   -
   -

2. c.add(1) gibt true zurück. Mögliche Erklärungen:
   -
   -

3. c.add(null); Mögliche Ergebnisse:
   -
   -

Aufgabe 2 – Lösungen

1. c.add(1) gibt false zurück. Mögliche Erklärungen:
   - Bei c handelt es sich um ein Set und eine 1 war bereits vorhanden.
   - Ein Bag würde mehrere 1 erlauben, darum nicht möglich.

2. c.add(1) gibt true zurück. Mögliche Erklärungen:
   -
   -

3. c.add(null); Mögliche Ergebnisse:
   -
   -

Aufgabe 2 – Lösungen

1. c.add(1) gibt false zurück. Mögliche Erklärungen:
   - Bei c handelt es sich um ein Set und eine 1 war bereits vorhanden.
   - Ein Bag würde mehrere 1 erlauben, darum nicht möglich.

2. c.add(1) gibt true zurück. Mögliche Erklärungen:
   - c instanceof Set: Eine 1 wurde hinzugefügt, war vorher noch nicht drin.
   - !(c instanceof Set): Eine 1 wurde hinzugefügt, nicht bekannt ob schon eine drin war.

3. c.add(null); Mögliche Ergebnisse:
   -
   -

Aufgabe 2 – Lösungen

1. c.add(1) gibt false zurück. Mögliche Erklärungen:
   - Bei c handelt es sich um ein Set und eine 1 war bereits vorhanden.
   - Ein Bag würde mehrere 1 erlauben, darum nicht möglich.

2. c.add(1) gibt true zurück. Mögliche Erklärungen:
   - c instanceof Set: Eine 1 wurde hinzugefügt, war vorher noch nicht drin.
   - !(c instanceof Set): Eine 1 wurde hinzugefügt, nicht bekannt ob schon eine drin war.

3. c.add(null); Mögliche Ergebnisse:
   - Falls null erlaubt: Null wird in die Sammlung aufgenommen, Rückgabewert analog zu oben.
   - Falls null nicht erlaubt: NullPointerException

Aufgabe 2 – Lösungen

4. c.remove(1); c.remove(1); gibt beide Male true zurück. Mögliche Erklärungen:
   -

5. Aufruf um zu prüfen, ob c mindestens ein Element 1 enthält:
   -

6. Programm zum Löschen aller Elemente 1 aus c, wenn gilt c instanceof Set:
   -

7. Programm zum Löschen aller Elemente 1 aus c, wenn gilt !(c instanceof Set):
   -

Aufgabe 2 – Lösungen

4. c.remove(1); c.remove(1); gibt beide Male true zurück. Mögliche Erklärungen:
   - Es gilt !(c instanceof Set). 1 konnte zweimal entfernt werden.

5. Aufruf um zu prüfen, ob c mindestens ein Element 1 enthält:
   -

6. Programm zum Löschen aller Elemente 1 aus c, wenn gilt c instanceof Set:
   -

7. Programm zum Löschen aller Elemente 1 aus c, wenn gilt !(c instanceof Set):
   -

Aufgabe 2 – Lösungen

4. c.remove(1); c.remove(1); gibt beide Male true zurück. Mögliche Erklärungen:
   - Es gilt !(c instanceof Set). 1 konnte zweimal entfernt werden.

5. Aufruf um zu prüfen, ob c mindestens ein Element 1 enthält:
   - c.contains(1);

6. Programm zum Löschen aller Elemente 1 aus c, wenn gilt c instanceof Set:
   -

7. Programm zum Löschen aller Elemente 1 aus c, wenn gilt !(c instanceof Set):
   -

Aufgabe 2 – Lösungen

4. c.remove(1); c.remove(1); gibt beide Male true zurück. Mögliche Erklärungen:
   - Es gilt !(c instanceof Set). 1 konnte zweimal entfernt werden.

5. Aufruf um zu prüfen, ob c mindestens ein Element 1 enthält:
   - c.contains(1);

6. Programm zum Löschen aller Elemente 1 aus c, wenn gilt c instanceof Set:
   - c.remove(1); Ein Set enthält maximal eine 1.

7. Programm zum Löschen aller Elemente 1 aus c, wenn gilt !(c instanceof Set):
   -

Aufgabe 2 – Lösungen

4. c.remove(1); c.remove(1); gibt beide Male true zurück. Mögliche Erklärungen:
   - Es gilt !(c instanceof Set). 1 konnte zweimal entfernt werden.

5. Aufruf um zu prüfen, ob c mindestens ein Element 1 enthält:
   - c.contains(1);

6. Programm zum Löschen aller Elemente 1 aus c, wenn gilt c instanceof Set:
   - c.remove(1); Ein Set enthält maximal eine 1.

7. Programm zum Löschen aller Elemente 1 aus c, wenn gilt !(c instanceof Set):
   - while (c.remove(1)) {}

Programmieren: Collections mit Array implementieren

		Sortierung	
		Sortiert	**Unsortiert**
Semantik	**Bag-Semantik**	SortedBag	UnsortedBag
	Set-Semantik	SortedSet	UnsortedSet

Programmieren: Collections mit Array implementieren

Gruppenarbeit (ca. 40 min)

Setzen Sie sich in der Gruppe zusammen.

Nehmen Sie das Blatt *Programmieren* und lösen Sie die Schritte 1 - 6 für ihre zugeteilte Collection. (Beginnen Sie mit Schritt 2)

Präsentieren Sie die Resultate aus Schritt 1 und Schritt 5 als Gruppe der Klasse.

Wichtig:
Jede Person programmiert am eigenen Computer!

Ziel der Übung:

- Implementieren einer eigenen Collection
- Abschätzen des Zeitaufwands für verschiedene Operationen
- Gedanken zu Einsatzzwecken

Collection als Array implementiert		**UnsortedBag**	**SortedBag**	**UnsortedSet**	**SortedSet**
	Inhalt Einfügen von: 12, 5, 28, 47, 28 und Löschen von: 28, 5	12, 5, 28, 47, 28 12, 47, 28	5, 12, 28, 28, 47 12, 28, 47	12, 5, 28, 47 12, 47	5, 12, 28, 47 12, 47
Asyptotische Komplexität im **Worst Case** (n Elemente in der Collection)	add(E e)	**O(1)** *Kein Suchen, kein Verschieben*	O(n)	O(n) *Suchen O(n) + Hinzufügen O(1)*	O(n)
	contains(Object o)	O(n) *Lineare Suche*	**O(log n)** *Binäre Suche*	O(n) *Lineare Suche*	**O(log n)** *Binäre Suche*
	remove(Object o)	O(n) *Suchen O(n) + Entfernen O(1)*	O(n)	O(n) *Suchen O(n) + Entfernen O(1)*	O(n)
	Besonders gut geeignet für:	*Häufiges Hinzufügen*	*Häufiges Suchen*	*Set Semantik, wenn für die Elemente keine Ordnungsrelation definiert ist*	*Set Semantik, wenn eine Ordnungsrelation definiert ist*

Hausaufgaben

1. Implementieren Sie Ihre Collection aus dem Unterricht fertig.
2. Implementieren Sie zusätzlich mindestens eine weitere Collection aus (SortedBag, UnsortedBag, SortedSet, UnsortedSet).