Fachhochschule
Nordwestschweiz

02 Listen

Algorithmen und Datenstrukturen 2

Themen Heute

Datenstrukturen

- Linked List
- Stack
- Queue

Java Collection Framework

Lernziele

Linked List

- Sie implementieren Linked Lists in Java.
- Sie kennen Vor- und Nachteile verschiedener Implementierungsvarianten (z.B. von add()).
- Sie setzen Linked Lists ein, um Datenstrukturen mit speziellen Zugriffsregeln zu implementieren (z.B. Stack und Queue).

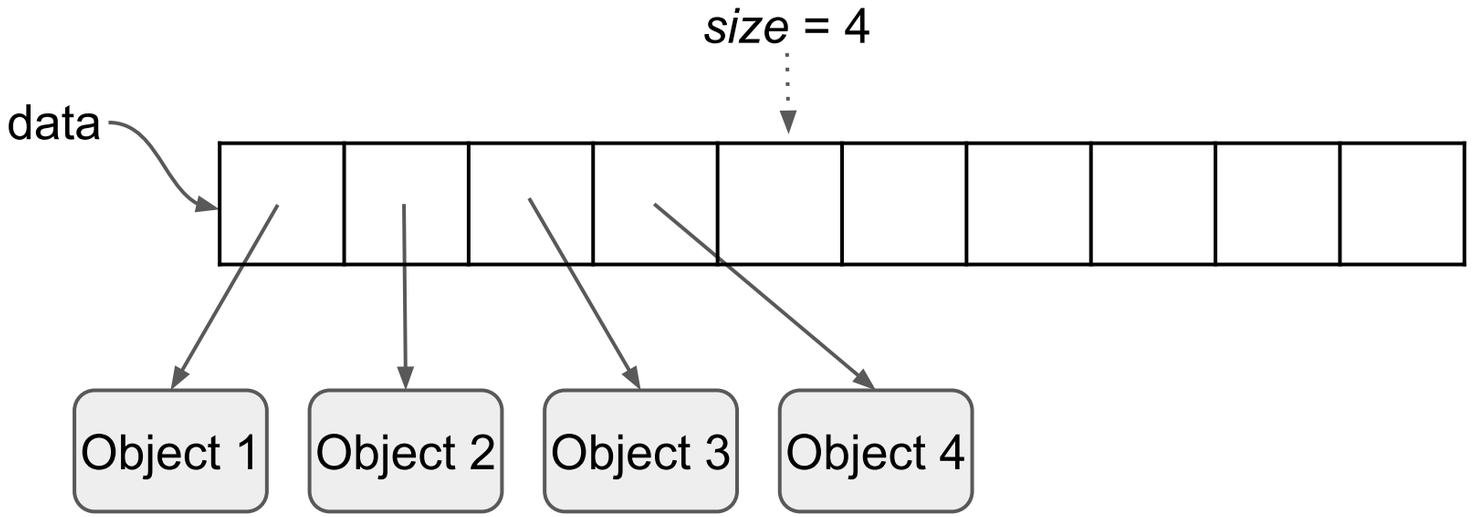
Collection Framework

- Sie nennen Vorteile und Nachteile der verschiedenen Collections und wägen diese anhand von gegebenen Anforderungen gegeneinander ab.

Array vs Linked List

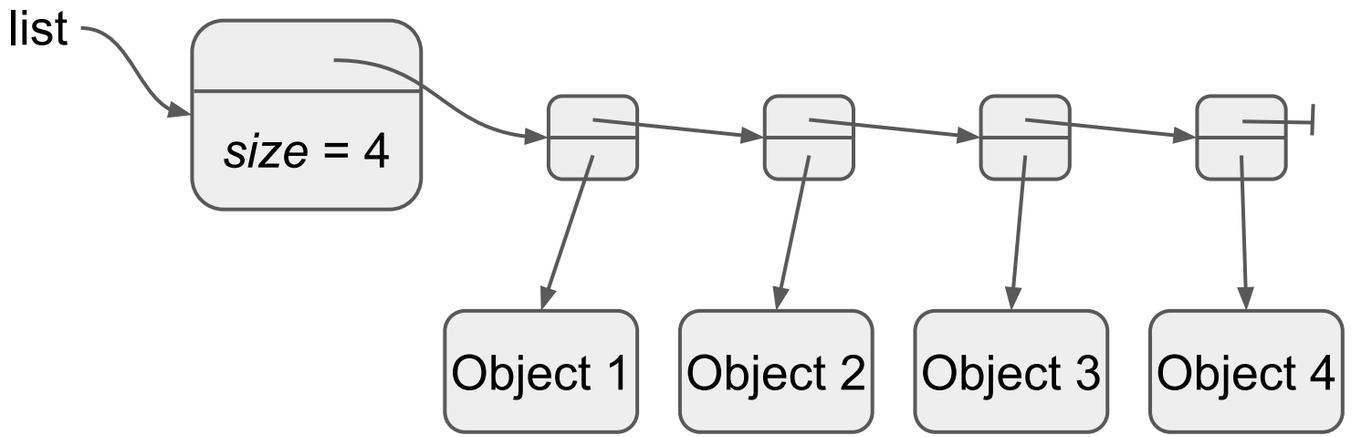
Collection als Array

- Fixe Grösse
- benötigt entweder zu viel Platz oder ist ständig voll

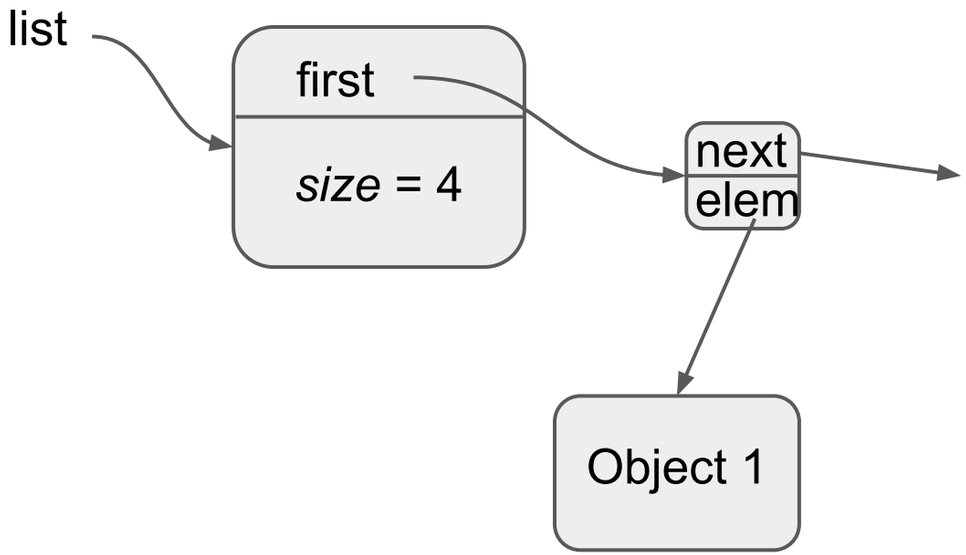


Collection als Linked List

- Dynamische Grösse
- benötigt nur so viel Platz wie Elemente in der Collection sind (plus O(1))

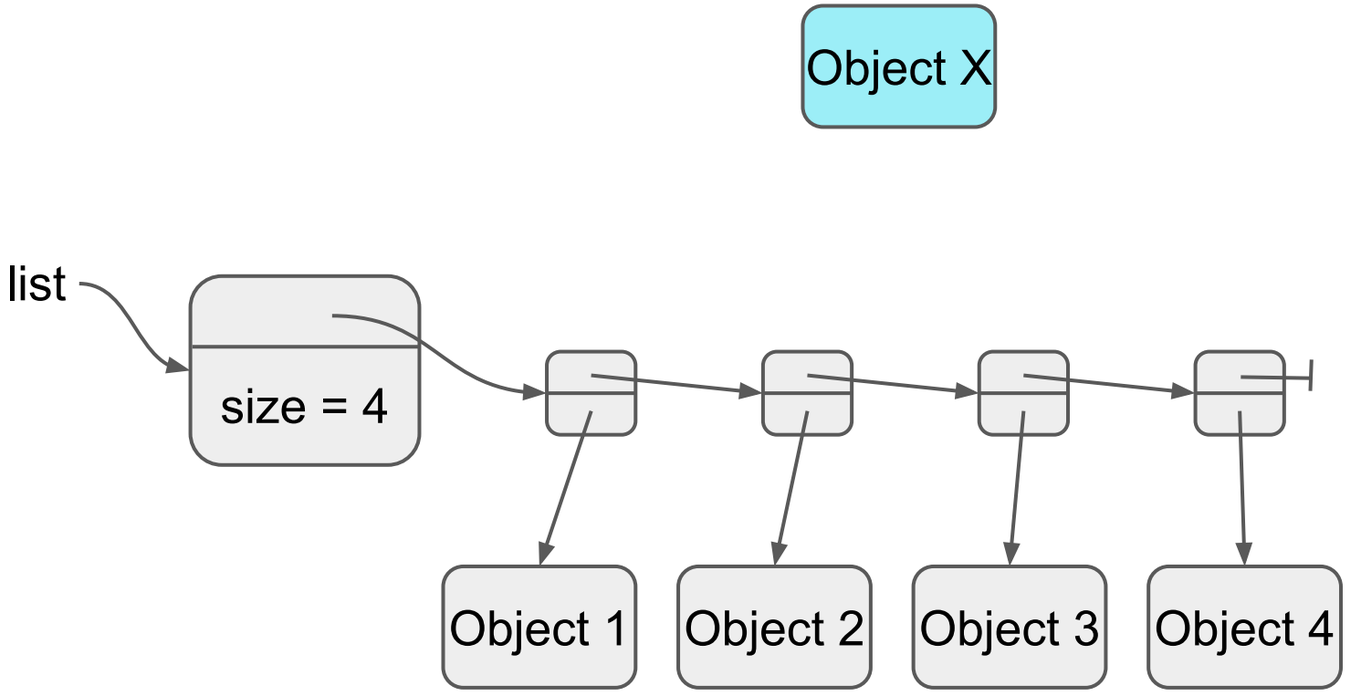


Linked List in Java



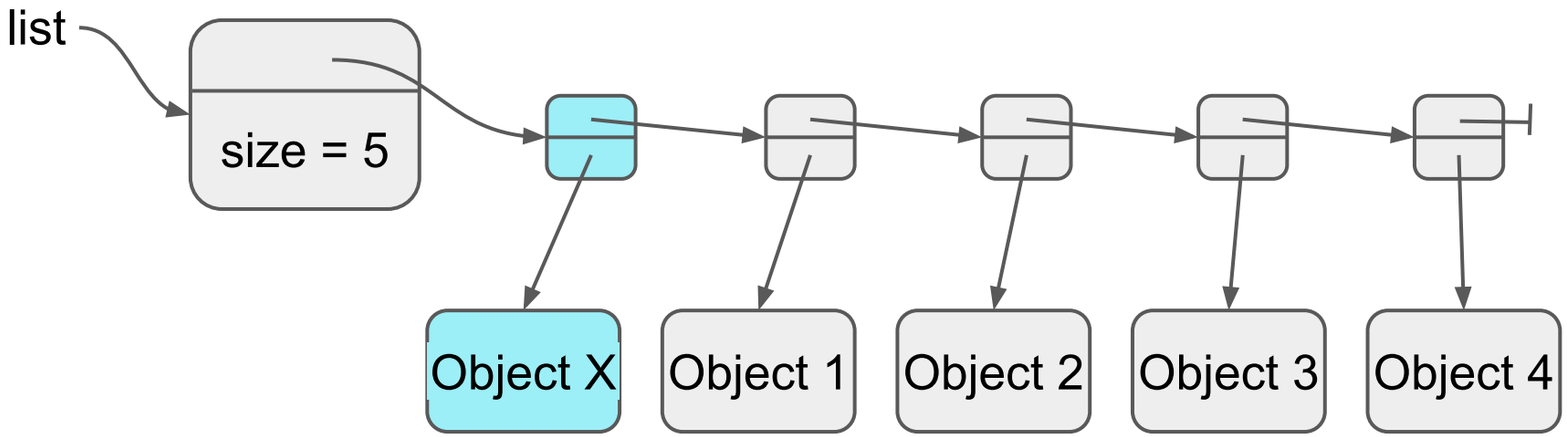

```
public class LinkedList<E> implements List<E> {
    private int size = 0;
    private Node<E> first;

    private static class Node<E> {
        private final E elem;
        private Node<E> next;

        private Node(E elem) { this.elem = elem; }
        private Node(E elem, Node<E> next) { this.elem = elem; this.next = next; }
    }
…
}
```

Linked List - Hinzufügen



Linked List - Hinzufügen am Anfang

Laufzeit: O(1)



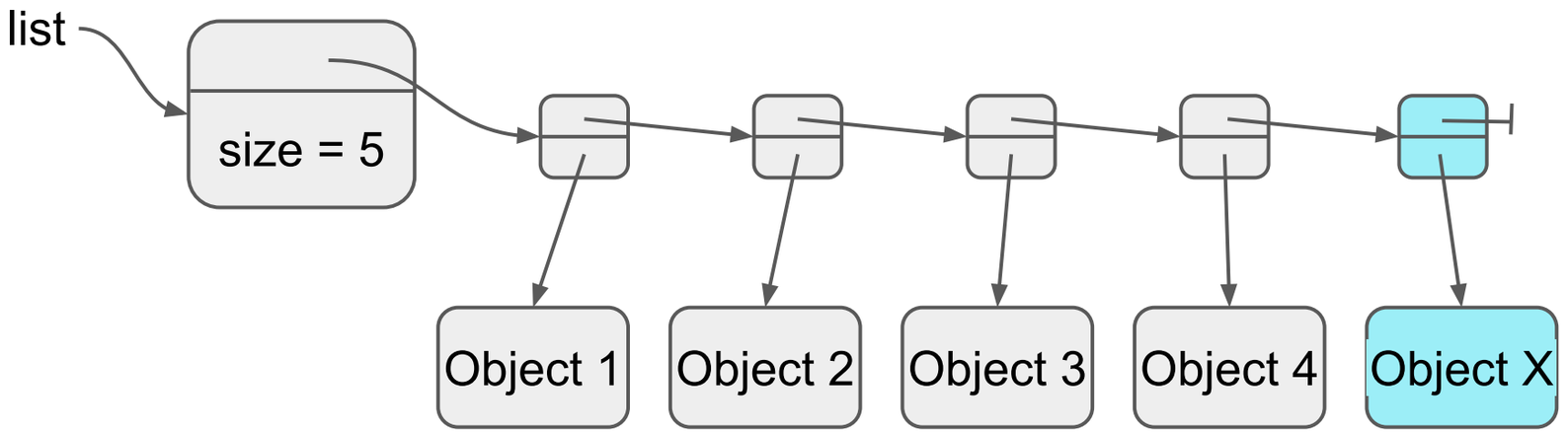
Linked List - Hinzufügen am Anfang

```
public boolean addHead(E e) {
    Node<E> node = new Node<E>(e, null);
    node.next = first;
    first = node;
    size++;
    return true;
}
```

Linked List - Hinzufügen am Ende

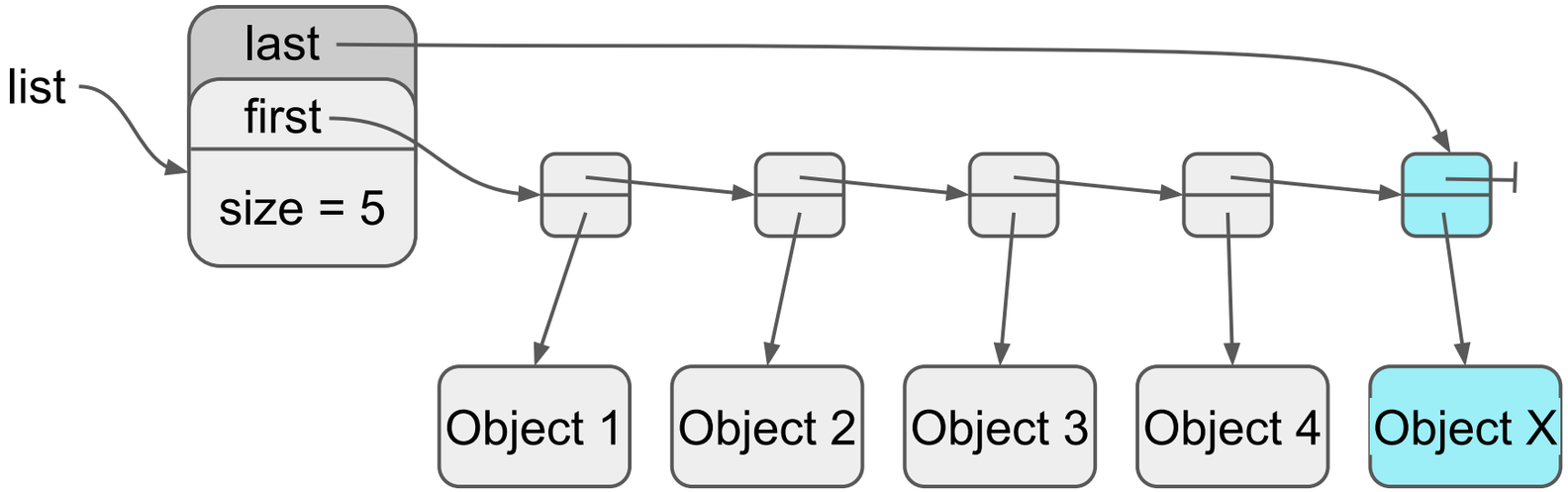
Laufzeit: O(n)

weil durch die ganze Liste
iteriert werden muss.



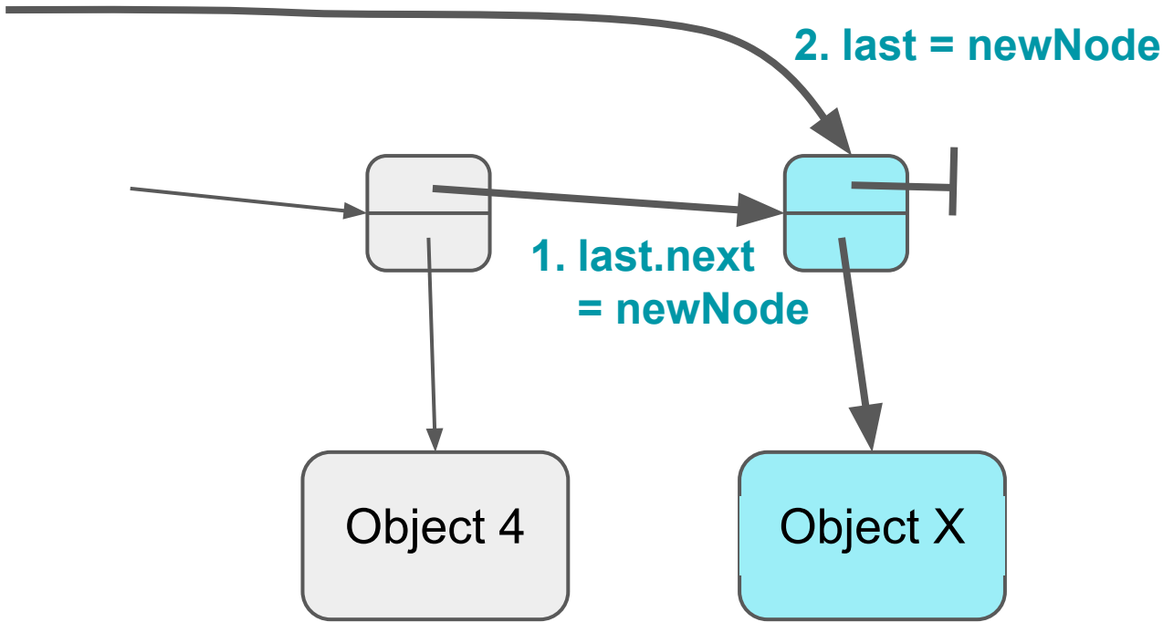
Linked List - Hinzufügen am Ende mit last Zeiger

Laufzeit: O(1)



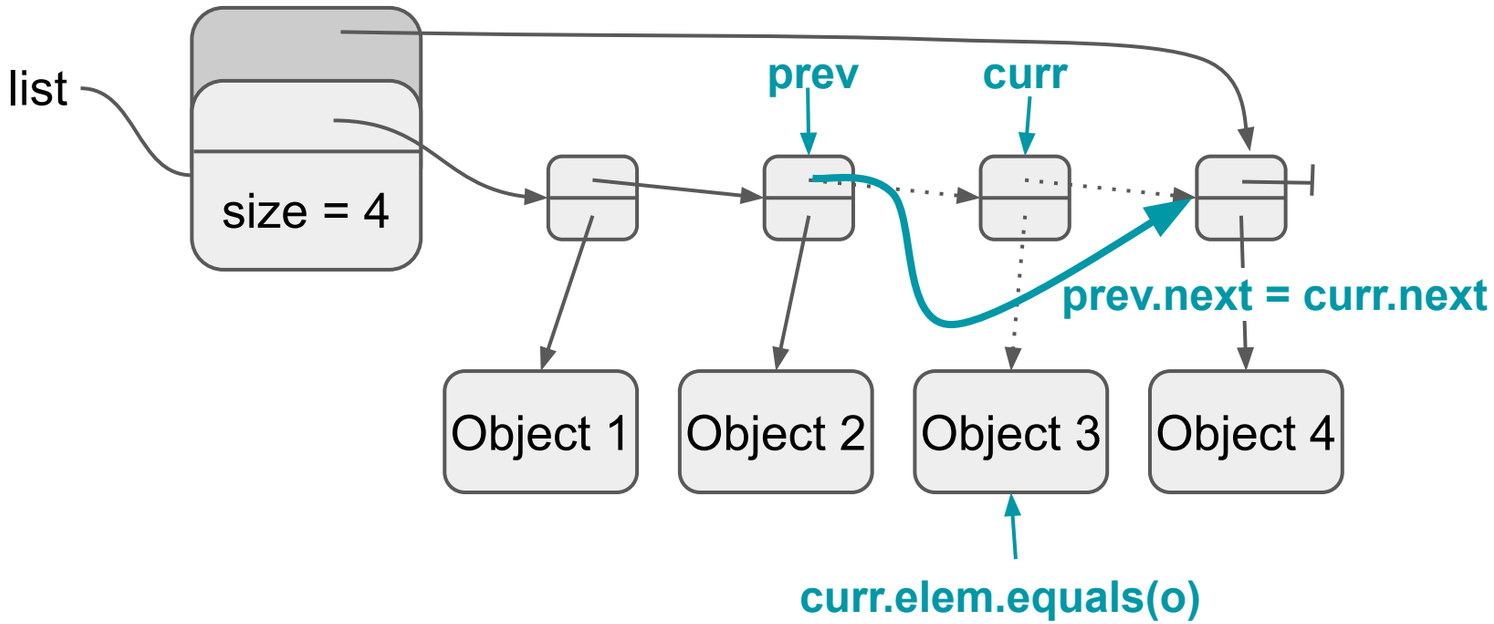
Linked List - Hinzufügen am Ende mit last Zeiger

Laufzeit: O(1)



Linked List - Entfernen

Laufzeit: O(n)



Linked List in Java

Implementieren Sie die Methoden (Programmieren Aufgaben 1A, 1B, 1C)

- `boolean add(E e) (Variante Hinzufügen am Ende mit Last-Zeiger)`
- `boolean remove(Object o)`
- `boolean contains(Object o)`

Denken Sie daran, den Last-Zeiger auch in der remove Methode zu aktualisieren.

Besprechen Sie sich gerne zu zweit, programmieren Sie selber!

Linked List in Java - boolean add(e E)

```
public class MyLinkedList<E> extends MyAbstractList<E> {
    private int size = 0;
    private Node<E> first;
    private Node<E> last;

    @Override
    public boolean add(E e) {
        Node<E> node = new Node<E>(e, null);
        if(last != null) { last.next = node; }
        else { first = node; }
        last = node;
        size++;
        return true;
    }
}
```

Linked List in Java - `boolean contains(Object o)`

```
@Override
public boolean contains(Object o) {
  Node<E> current = first;
  while (current != null && !Objects.equals(o, current.elem)) {
     current = current.next;
  }
  return current != null;
}
```

Linked List in Java - `boolean remove(Object o)`

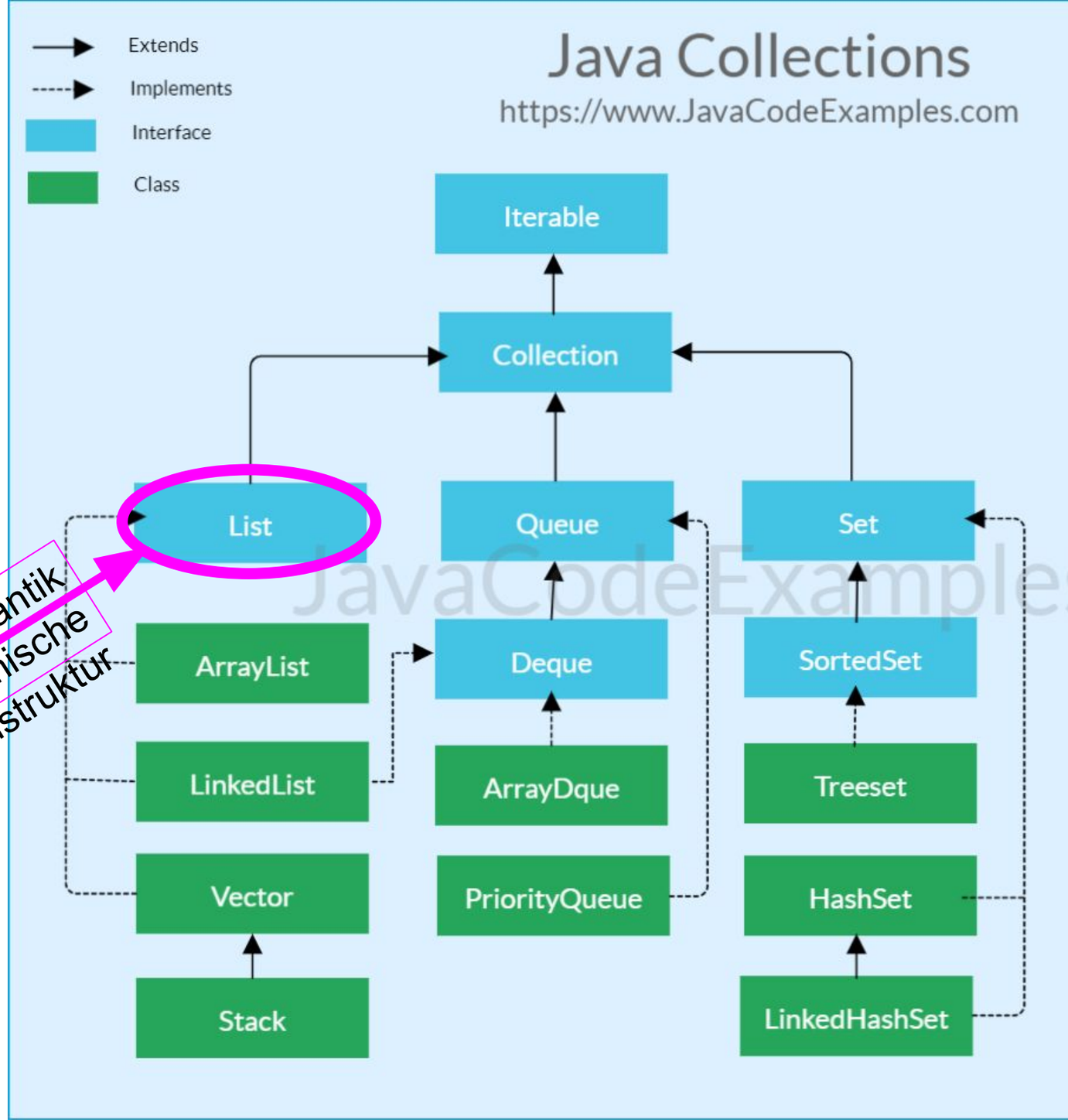
```
@Override
public boolean remove(Object o) {
  Node<E> current = first, prev = null;
  while (current != null && !Objects.equals(o, current.elem)) {
     prev = current;
     current = current.next;
  }
  if (current != null) {
     if (current == last) last = prev;
     if (current == first) first = current.next;
     else prev.next = current.next;
     size--;
     return true;
  } else return false;
}
```

List Interface im Java Collection Framework

Standardoperationen

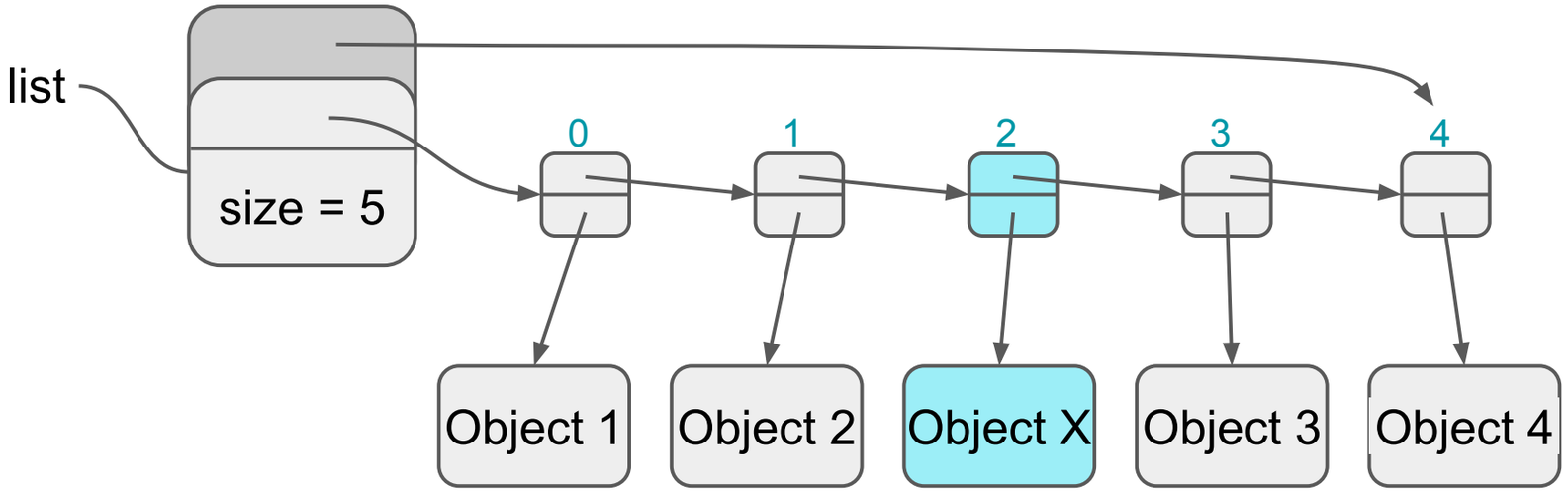
- boolean add(E e) **<- Am Ende der Liste!!**
- boolean remove(Object o)
- boolean contains(Object o)

- void add(int index, E e)
- E get(int index)
- E remove(int index)



Linked List - Hinzufügen an Index 2

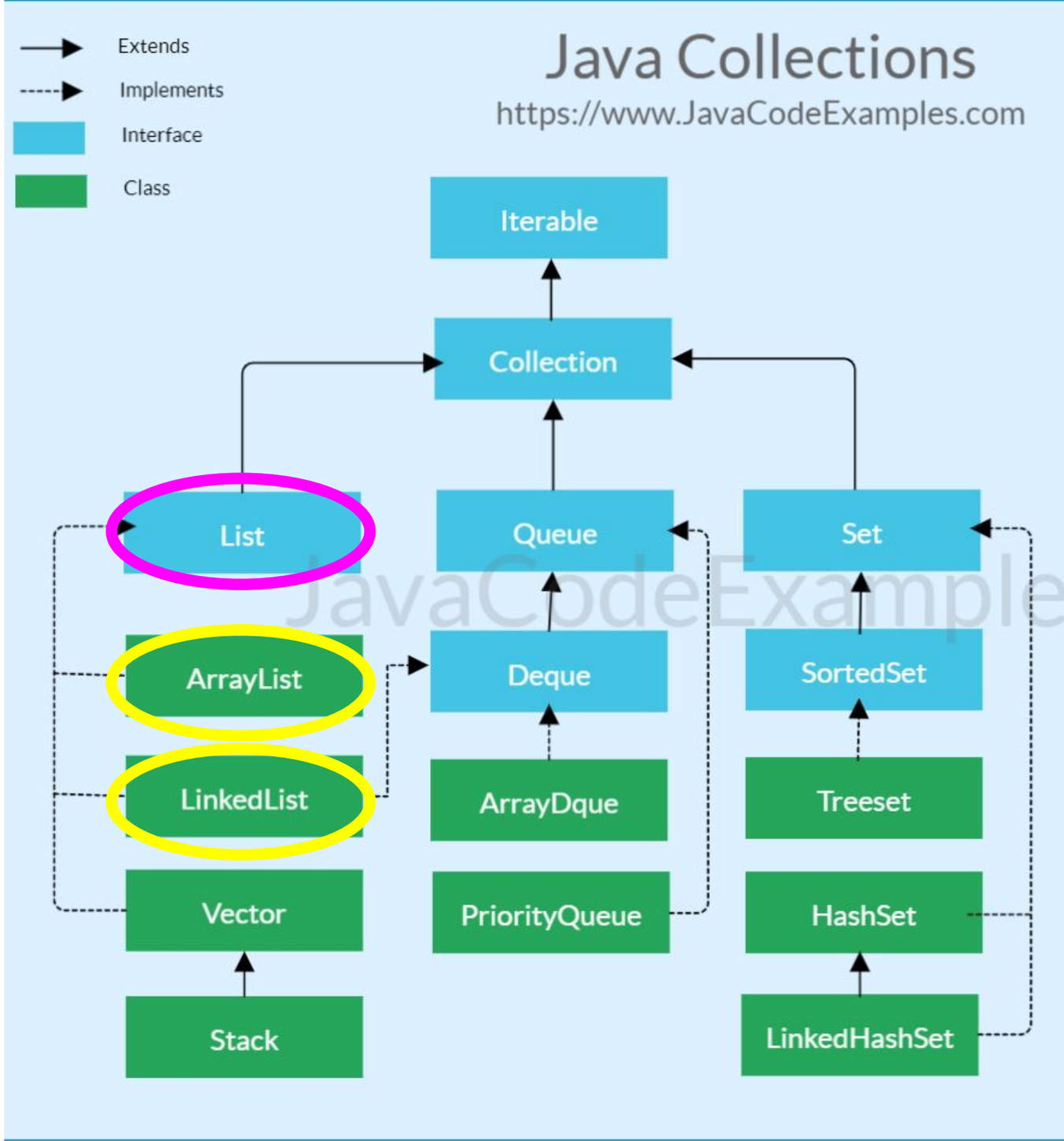
Laufzeit: O(n)



List Interface im Java Collection Framework

Vorhandene Implementierungen des List Interface (Bag Semantik & unsortiert)

- Class LinkedList
  - Zugriff auf Index in O(n)
- Class ArrayList
  - unbegrenzte Länge
    wird automatisch vergrössert
    und verkleinert
  - Zugriff auf Index in O(1)
  - Aufwand remove in O(n)



Stack

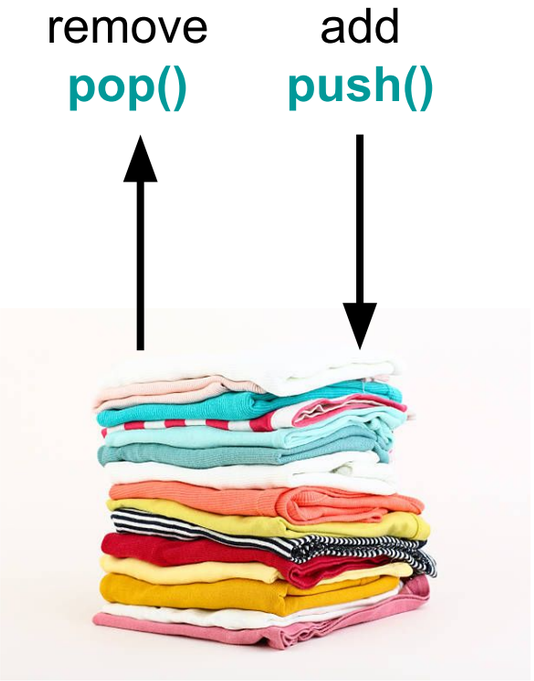
LIFO - Last In First Out

Methoden u.a.

- push(), pop()
- peek() gibt Referenz auf oberstes Element
  auch top() in Literatur

Java Collection Framework

- Class Stack implementiert List Interface
  https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Stack.html



Queue

FIFO - First In First Out

Methoden u.a.

- add(), offer() in Java / enqueue() in Literatur
- remove(), poll() in Java / dequeue() in Literatur
- Referenz auf erstes Element: element(), peek()

Java Collection Framework

- Interface Queue
  https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Queue.html

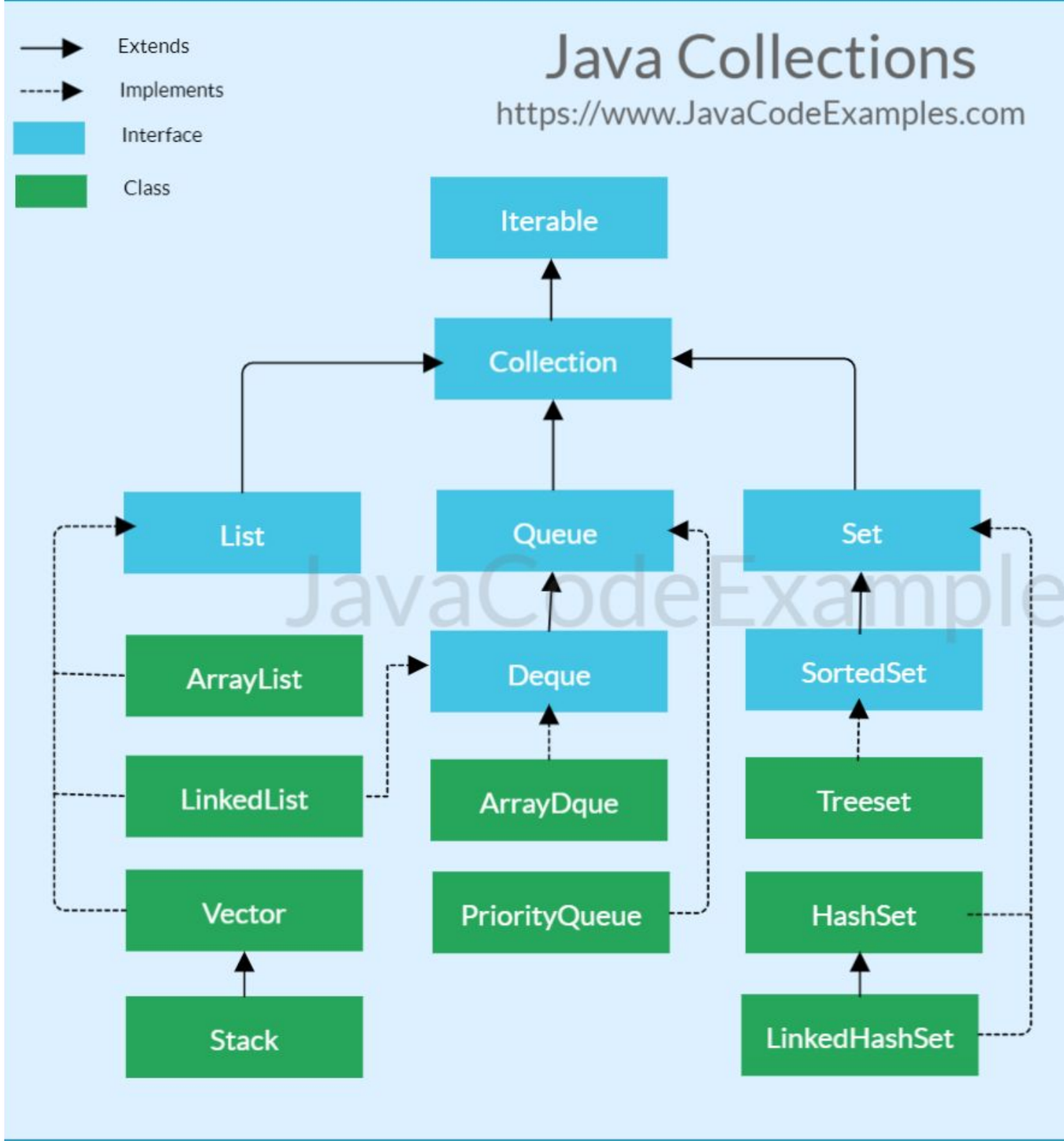
Takehome

Collection ist eine Sammlung von Elementen derselben Klasse.

Das **Collection Interface** kennt Methoden zum Einfügen, Löschen, Suchen

List (dynamisch, indexiert),
Set (keine Duplikate),
Queue (FIFO),
Stack (LIFO) sind spezielle Collections

LinkedList und **ArrayList** implementieren das List Interface



Hausaufgaben

Programmieren

- Aufgabe 1D
- Aufgabe 1E (optional)
- Aufgabe 2

Arbeitsblatt