University of Basel

Programmieren I

Block 9 - Einführung

Andreas Morel-Forster, Departement Mathematik und Informatik, Universität Basel

Wo stehen wir?

- Einfaches Bild eines Computers (CPU, Speicher, Bus)
- Variablen, Daten und Typen
- Strukturiertes Programmieren
  - Ausdrücke und Anweisungen
  - Sequenzen von Anweisungen
  - Verzweigungen
  - Schleifen
  - Methoden und Funktionen
- Arrays und Strings
- Primitive und Referenz-Datentypen
- Klassen und Objekte
- Vererbung und Override (Interfaces, abstrakte Klassen)

Es fehlt noch: Routine

Every great developer you know got there by solving problems they were unqualified to solve until they actually did it.

-Patrick McKenzie

Was kommt heute?

- Vererbung: Mini-Übung Nachbesprechung
- Java-Files: Struktur und Organisation
- Dynamische Datenstrukturen
- Prüfung: Informationen & Test der Umgebung
- Übungen Besprechen & Offene Frage-Antwort Session

Vererbung – Besprechung Mini-Übung

- Jupyter-Notebook

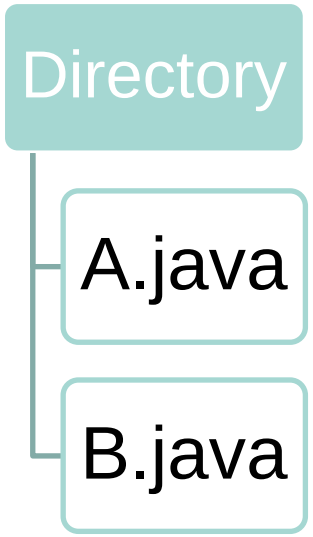
Java Files: Struktur

MyProgramm.java

```
import java.Math; // imports zuoberst

public class MyProgramm { // Klasse zuäusserst

        int count; // Variablen

        MyProgramm() {} // Konstruktoren

        static void f() { // Methoden
                System.out.println("some");
        }

        // main Methode – Start des Programms
        public static void main(String[] args) {
                f();
        }
}
```

Java Files: Organisation

Alles im selben Directory funktioniert ohne weiteres zutun.



A.java

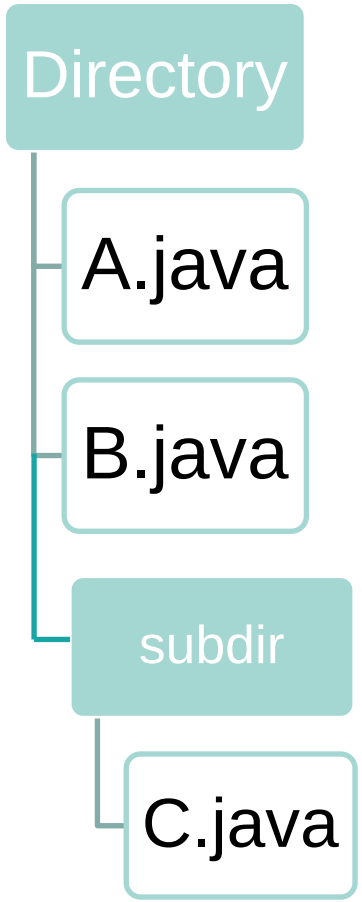
```
public class A {
        public static void main(String[] args) {
                B b = new B();
                b.f();
                B.g();
        }
}
```

B.java

```
public class B {
        int x;
        int y;

        void f() { /* … */ }
        static void g() { /* … */ }
}
```

Java Files: Organisation (nicht Prüfungsrelevant: packages)

Wenn ein Konstruktor in C definiert wird, dann muss der public sein um in B verwendet werden zu können.
Mehr Details in Programmierung 2.



A.java

```
public class A {
        public static void main(String[] args) {
                B b = new B();
        }
}
```

B.java

```
import subdir.C;

public class B {
        C c;
        B() {
                c = new C();
        }
}
```

C.java

```
package subdir;

public class C {
        public C() { }
}
```

Java Files: Organisation

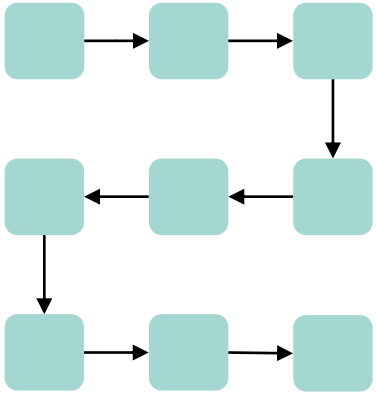
Demo – VS Code

Dynamische Datenstrukturen - Motivation

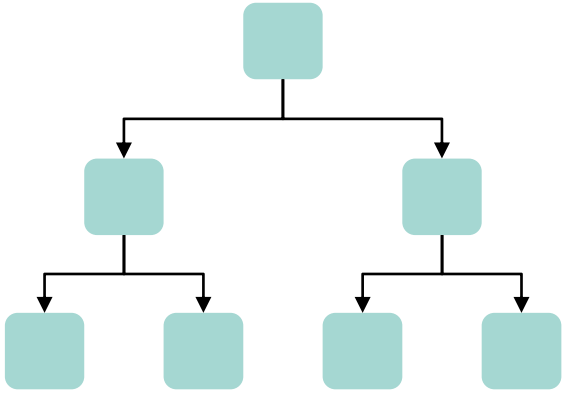
Arrays sind nicht immer gut geeignet:

- Was wenn Anzahl Elemente nicht bekannt ist?
- Einfügen in ein Array kann "teuer" sein.
- Was wenn ein Element hinzu kommt wenn das Array schon voll ist?
- Lineare Struktur nicht immer das richtige

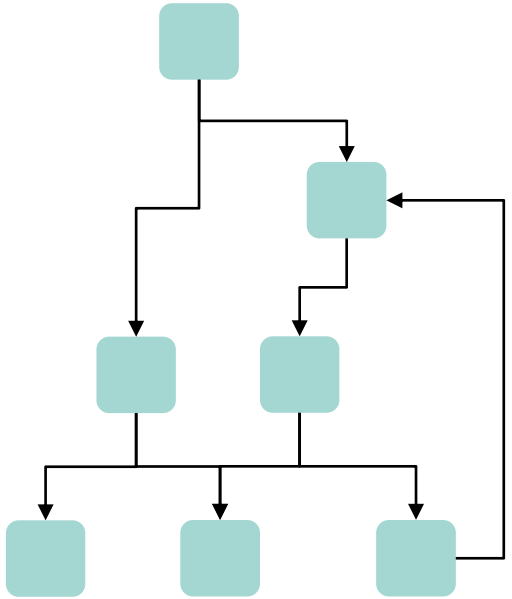
Flexiblere "Struktur" wünschenswert.



linear Liste



Baumstruktur



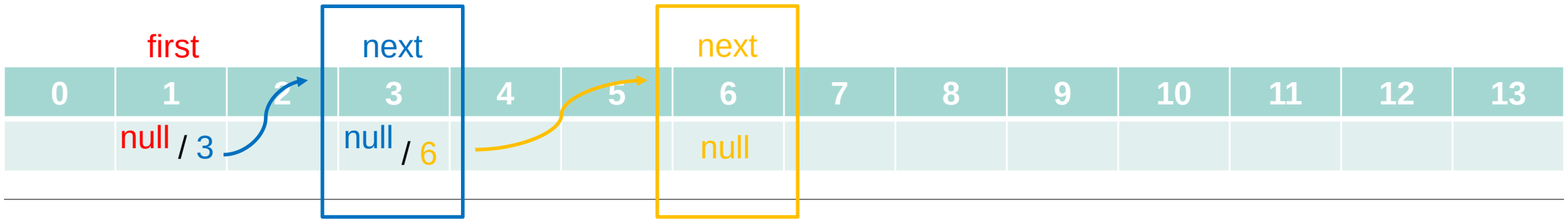
Graphen

Dynamische Datenstrukturen - Prinzip

Objekte können Adressen von anderen Objekten speichern, auch der eigenen Klasse.

```
public class Node {
    Node next;

    public static void main(String[] args) {
        Node first;
        first = new Node();
        first.next = new Node();
    }
}
```

Variable	Ort	Wert	Objekt
first	1	3	blau
first.next	3	6	gelb
first.next.next	6	null	(keines)



Dynamische Datenstrukturen - Liste

Demo in VS Code

1. Liste von Zahlen
2. Einfügen (am Anfang und am Ende)
3. Sortierte Liste (sortiertes Einfügen)

je nach Zeit: Einkaufsliste (Vererbung)

Prüfung

Prüfung

Termin

- Datum **8. Januar 2021**
- *(Termin im November oder Dezember wurde nicht genehmigt.)*

Prüfungsart

- Entscheidung vom Programmkomitee noch ausstehend.

Prüfung Vorbereiten

- Konzepte in den Jupyter-Notebooks nachvollziehen
- Mini-Übungen in Jupyter-Notebooks lösen
- Einfache Übungen von "Sprechen Sie Java", Liste auf der Vorlesungsseite
- Fortgeschritten: Übungsblätter lösen
- Eigene Programme umsetzen

Betreuung bis zur Prüfung

- Forum wird weiterhin betreut, wahrscheinlich mit längeren Antwortzeiten
- Wenn gewünscht, mögliche online Fragestunde (Ende November / Anfang Dezember ?)

Prüfungs Themen

Kennen und Anwenden

- Variablen und Zuweisungen
- Kommentare
- Einfache Datentypen
- Verzweigungen
- Vergleichsoperatoren
- Schleifen
- Methoden und Funktionen
- Lokale und statische Variablen
- Sichtbarkeit und Lebensdauer von Variablen
- Arrays
- Zeichen und Strings
- Klassen und Objekte
- Vererbung und Überschreiben

Kennen, aber nicht anwenden

- Interfaces
- Abstrakte Klassen

Nicht relevant:

- Turtlegrafik
- Computerarchitektur
- AWT/Swing (Übungen)
- Packages
- Gradlew
- Java API
  (wenn wird die Funktion angegeben)

Arten von Prüfungsfragen (Auswahl, nicht komplett)

Schreiben Sie ….

- eine Variabledefinition für …
- einen boolschen Ausdruck der…
- eine Funktion welche …
- eine Klasse um ….

Gegeben sei der folgende Code: ….

- Was tut der Code? Was wird ausgegeben?
- Weshalb tut der Code nicht was er soll?
- Weshalb kompiliert der Code nicht? Und wie müsste man den Code korrigieren?
- Terminiert die Schleife? Wie oft wird die Schleife ausgeführt?
- Tun beide Code-Fragmente das selbe?

Bewerten Sie folgende Aussage über Java: ….

Ordnen Sie die folgenden Aussagen den entsprechenden… Funktionen, Variablen, Codestücken … zu.

Wandeln Sie den folgenden Code um …. for-Schleife <-> while-Schleife ….

Demo-Prüfung

- Testen des Systems
- Testen der Art der Aufgaben
- Schwierigkeit nicht exemplarisch
- "Zeitdruck" nicht exemplarisch